UNIVERSIDADE
FEDERAL RURAL
DE PERNAMBUCO

Annelyelthon Ferreira Alves

Aplicação de Métodos Ágeis em Desenvolvimento Global de Software

Recife

2021

Annelyelthon Ferreira Alves

Aplicação de Métodos Ágeis em Desenvolvimento Global de Software

Monografia apresentada ao Curso de Bacharelado em Ciências da Computação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Ciências da Computação.

Universidade Federal Rural de Pernambuco – UFRPE

Departamento de Computação

Curso de Bacharelado em Ciências da Computação

Orientador: Marcelo Luiz Monteiro Marinho

Recife

2021

Dados Internacionais de Catalogação na Publicação
Universidade Federal Rural de Pernambuco
Sistema Integrado de Bibliotecas
Gerada automaticamente, mediante os dados fornecidos pelo(a) autor(a)

A474a Alves, Annelyelthon Ferreira
Aplicação de Métodos Ágeis em Desenvolvimento Global de Software / Annelyelthon Ferreira Alves. - 2021.
83 f. : il.

Orientador: Marcelo Luiz Monteiro Marinho.
Inclui referências, apêndice(s) e anexo(s).

Trabalho de Conclusão de Curso (Graduação) - Universidade Federal Rural de Pernambuco, Bacharelado em Ciência da Computação, Recife, 2021.

1. Desenvolvimento de software global. 2. Desenvolvimento ágil de software. 3. Revisão sistemática da literatura. 4. Engenharia de software. I. Marinho, Marcelo Luiz Monteiro, orient. II. Título

CDD 004

MINISTÉRIO DA EDUCAÇÃO E DO DESPORTO
UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO (UFRPE)
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

http://www.bcc.ufrpe.br

Ficha de Aprovação do Trabalho de Conclusão de Curso

Trabalho defendido por Annelyelthon Ferreira Alves às 08 horas do dia 22 de julho de 2021, no link https://meet.google.com/qei-garr-cuv, como requisito para conclusão do curso de Bacharelado em Ciência da Computação da Universidade Federal Rural de Pernambuco, intitulado Aplicação de Métodos Ágeis em Desenvolvimento Global de Software, orientado por Marcelo Marinho e aprovado pela seguinte banca examinadora:

Marcelo Marinho
DC/UFRPE

Suzana Sampaio
DC/UFRPE

Agradecimentos

Gostaria de começar meus agradecimentos falando um pouco sobre a minha trajetória até este momento. No ano de 2016, influenciado pelo meu primeiro “emprego” como jovem aprendiz no setor de TI, e por ter afinidade com a área, escolhi Bacharelado em Ciência da Computação como primeira opção na Universidade Federal Rural de Pernambuco, depois de conseguir uma boa nota no ENEM. Ao iniciar o curso, percebi que não era como eu imaginava, era muito mais complexo do qualquer coisa que eu já tinha visto no ensino médio. Desta forma, os primeiros períodos foram os mais complicados, entretanto, creio que esta fase foi mais facilmente superada porque me unir com as pessoas certas desde o início, Eder Lucena e Fábio Alves, foram pessoas às quais me indifiquei, nós sempre nos reuníamos para estudar as disciplinas que pagavamos juntos, e em meio a muitas risadas e diversão, aprendiamos muito, eles foram colegas de turma, que ao longo do tempo, pude ter o prazer de chama-los de amigos. São as pessoas mais importantes a quem dedico estes agradecimentos neste documento.

Além destes, gostaria de agradecer a algumas pessoas desta universidade que foram responsáveis pelo minha formação acadêmica, e que tenho admiração. Marcelo Marinho, foi meu orientador da Iniciação científica, PCC e TCC, foi quem me deu todo suporte para publicação de um artigo científico e preparação dos meus documentos de conclusão de curso. Lorena Brizza é a única professora desta lista, que não pertence ao departamento de computação, mas sim ao departamento de matemática, foi com ela que consegui de fato aprender Cálculo, e em questão de didática, acredito ser a melhor professora que tive neste curso. Vanilson Burégio foi meu professor de Programação II e Desenvolvimento Web, devo muito dos meus conhecimentos destas disciplinas a ele, além de ter sido um ótimo coordenador também. Rafael Dueire dispensa comentários, ele me ensinou Compiladores e foi o professor mais comprometido que eu já vi na minha vida, nunca sequer se atrasou, que dirá faltar algum dia, além de ter um conhecimento surpreendente das cadeiras que ministrava. Fernando Aires foi meu professor em Sistemas Distribuídos e Segurança de Informação, suas aulas foram as mais divertidas de todas, sem faltar conhecimento, é claro. Sandra Xavier, nossa querida secretária de BCC, sempre me respondeu com paciência e rapidez qualquer tipo de burocracia que eu precisava resolver. Ruan Carvalho pelo excelente coordenador que é, ajudando não só a mim, mas também aos meus amigos com agilização de processos nesta reta final do curso.

A todos vocês, meu muito obrigado.

Resumo

O Desenvolvimento de Software Global (*Global Software Development* - GSD) continua crescendo substancialmente e está rapidamente se tornando um padrão, fundamentalmente diferente do desenvolvimento local da Engenharia de Software. Com o desenvolvimento ágil de software (*Agile Software Development* - ASD), tornou-se uma escolha atraente para empresas que tentam melhorar seu desempenho, embora seus métodos tenham sido originalmente projetados para equipes pequenas e individuais. A literatura atual não fornece uma imagem coesa de como as práticas ágeis são levadas em consideração na natureza distribuída do desenvolvimento de software: como fazer, quem envolver e o que funciona na prática. Este estudo tem como objetivo destacar como as práticas de ASD são aplicadas no contexto do GSD, a fim de desenvolver um conjunto de técnicas que podem ser relevantes tanto na pesquisa quanto na prática. Para responder à pergunta de pesquisa, “Como as práticas ágeis são adotadas em equipes de desenvolvimento de software ágil global?” Conduzimos uma revisão sistemática da literatura e um survey com engenheiros de software no contexto de ASD e GSD. Uma síntese das soluções encontradas em setenta e seis estudos forneceu 48 práticas distintas que as organizações podem implementar, incluindo “colaboração entre equipes”, “arquitetura ágil”, “coaching”, “demonstração do sistema” e “automação de teste”. Assim, realizamos uma verificação com engenheiros de software e identificamos que essas práticas implementáveis fornecem soluções para gerenciar equipes GSD e assim, abraçar a agilidade.

Palavras-chave: Desenvolvimento de software global; Desenvolvimento ágil de software; Revisão sistemática da literatura; Engenharia de software.

Abstract

Global Software Development (GSD) continues to grow and is rapidly becoming a standard, fundamentally different from local Software Engineering development. Withal, agile software development (ASD) has become an appealing choice for companies attempting to improve their performance although its methods were originally designed for small and individual teams. The current literature does not provide a cohesive picture of how the agile practices are taken into account in the distributed nature of software development: how to do it, who, and what works in practice. This study aims to highlight how ASD practices are applied in the context of GSD in order to develop a set of techniques that can be relevant in both research and practice. To answer the research question, ”how are agile practices adopted in agile global software development teams?” We conducted a systematic literature review and a survey with practitioners of the ASD and GSD literature. A synthesis of solutions found in seventy-six studies provided 48 distinct practices that organizations can implement, including ”collaboration among teams”, ”agile architecture”, ”coaching”, ”system demo” and ”test automation”. These implementable practices go some way towards providing solutions to manage GSD teams, and thus to embrace agility.

Keywords: Global Software Development; Agile Software Development; Systematic Literature Review; Software engineering.

Lista de ilustrações

Lista de abreviaturas e siglas

GSD	Global Software Development
ASD	Agile Software Development
AGSD	Agile Global Software Development
SLR	Systematic Literature Review
IEEE	Instituto de Engenheiros Elétricos e Eletrônicos
ACM	Aluminium Composite Material
DPP	Distributed Pair Programming
TDD	Test Driven Development
PO	Product Owner

Sumário

1 Introdução

Duas tendências fortes moldam a engenharia de software moderna. Em primeiro lugar, as empresas são fortemente obrigadas a embarcar em esforços de Desenvolvimento de Software Global (*Global Software Development - GSD*) para, entre outras razões, tentar reduzir o custo de produção e aumentar o *pool* de indivíduos talentosos disponíveis (HILLEGERSBERG; LIGTENBERG; AYDIN, 2011). Em segundo lugar, o Desenvolvimento Ágil de Software (*agile software development - ASD*) é cada vez mais aplicado em projetos (distribuídos) para suportar o manuseio mais dinâmico dos produtos (HILLEGERSBERG; LIGTENBERG; AYDIN, 2011).

O desenvolvimento Ágil e Global de Software (AGSD) é atualmente uma tendência significativa (VALLON et al., 2017). O 14º anual do relatório do estado ágil - VersionOne, afirma que 71% dos entrevistados disseram que sua organização pratica *Agile* com várias equipes co-localizadas colaborando além das fronteiras geográficas (VersionOne, Inc., 2020). O AGSD leva a muitos desafios, dada a natureza inerente de ambos os paradigmas: ASD e GSD. Por um lado, a comunicação GSD é comumente baseada em documentos, ou seja, conhecimento explícito que diminui o efeito das quatro distâncias desse paradigma (física, temporal, linguística e cultural) (MODI; ABBOTT; COUNSELL, 2017). Por outro lado, o manifesto ágil (BECK et al., 2001) afirma que, em ASD, as interações face-a-face são preferíveis do que seguir processos de comunicação restritos, além disso, é preferível a documentação abrangente (LOUS et al., 2018b; ALTAF et al., 2019).

Isso levanta uma questão prática para a pesquisa: **como as práticas ágeis são adotadas nas equipes globais de desenvolvimento de software ágil?** Para isso, conduzimos uma Systematic Literature Review (SLR) (KITCHENHAM; CHARTERS, 2007) que se concentrou em encontrar abordagens de pesquisa de 2001 a 2020 que destacaram a adaptação do Agile em todo o mundo.

Este trabalho de conclusão de curso contribui para o corpo de conhecimento do AGSD, apresentando um conjunto de práticas do AGSD. A fim de fornecer orientações concretas para projetos que desejam empregar o contexto de AGSD, descrevemos o *objetivo*, *como* ele pode ser adotado e *quem* é o responsável por realizar a prática.

Este trabalho de conclusão de curso está organizado da seguinte forma: no Capítulo 2, apresentamos o background do problema e definimos nossas questões de pesquisa. O Capítulo 3 descreve o método que aplicamos. O Capítulo 4 apresenta os resultados, suas implicações e limitações, respectivamente. Além disso, o Capítulo 5 apresenta discussão e limitações. Finalmente, o Capítulo 6 apresenta as conclusões

e direções de pesquisas futuras.

1.1 Objetivos:

1.1.1 Objetivo Geral:

Analisar as principais práticas consideradas ágeis bem como as mesmas são aplicadas dentro do contexto distribuído globalmente.

1.1.2 Objetivos Específicos:

1. Determinar o estado da arte do tema supracitado.
2. Fornecer uma compreensão abrangente e análise do uso de práticas ágeis em GSD.
3. Avaliar às práticas ágeis GSD com os engenheiros de software, visando garantir que o conhecimento adquirido representa de forma adequada, a visão e a perspectiva dos que desenvolvem software.

2 Referencial Teórico

2.1 Desenvolvimento Ágil de Software (ASD)

Segundo (PONTES; ARTHAUD, 2018), os métodos ágeis são uma coleção de práticas de desenvolvimento, que se adaptaram ao longo do tempo para que o desenvolvimento se torne muito mais rápido, com menor custo e com mais qualidade. Além disso, mudanças no meio dos projetos sempre ocorrem, e os métodos ágeis enxergam isso como uma vantagem, uma vez que elas são adaptivas, os desenvolvedores estaram sempre preparados para mudanças que não foram possíveis observar no início do projeto.

Em (DINGSøYR et al., 2017) foi dito que a definição de desenvolvimento ágil é como uma mudança de paradigma na engenharia de software, além de que os métodos ágeis transformaram a prática de desenvolvimento de software, promovendo entrega evolutiva e envolvimento ativo com usuário final. O sucesso dos métodos ágeis foi rapidamente repercutido e foi inicialmente utilizado por equipes pequenas que trabalhavam no mesmo local (DINGSøYR et al., 2017).

(JHA; VILARDELL; NARAYAN, 2016) descreve que a abordagens de desenvolvimento ágil de software procura dominar qualquer tipo de restrições do desenvolvimento de software orientado a planos, durante todas as fases de desenvolvimento do produto, sem deixar de faltar organização e agilidade, com isso, se faz possível atender as mudanças as necessidades do mercado.

2.2 Desenvolvimento de Software Global (GSD)

Desenvolvimento de Software Global (GSD) é o nome dado para designar empresas que distribuem seu desenvolvimento de software além de suas fronteiras nacionais (MARINHO; NOLL; BEECHAM, 2018). A maioria dessas empresas usa GSD em grande escala com o objetivo de obter cortes de despesas, vantagens econômicas, acesso a talentos globais, entrega mais rápida e um ciclo de desenvolvimento de software de 24 horas que é possível devido aos diferentes fusos horários (RIZVI; BAGHERI; GASEVIC, 2015). No entanto, embora a adoção do GSD tenha aumentado nas empresas, ter pessoas em locais diferentes pode levar a muitos problemas relacionados à comunicação, coordenação e controle do processo de desenvolvimento (HOSSAIN; BABAR; PAIK, 2009).

A principal diferença entre equipes globais e equipes co-localizadas é o fato

de que os membros da equipe não estão fisicamente presentes no mesmo ambiente ao mesmo tempo. Porém, ambas as equipes têm os mesmos objetivos que são desenvolver e entregar valor constante a um projeto ou produto. No entanto, o fato de os membros da equipe estarem espalhados em outros países leva a muito mais complicações do que os rostos de equipes co-localizadas, como comunicação assíncrona, falta de comunicação face-a-face, dificuldade de construir uma visão compartilhada com a equipe, difícil organizar reuniões devido aos diferentes fusos horários e diferentes níveis de conhecimento sobre uma linguagem comum (MARINHO; NOLL; BEECHAM, 2018).

2.3 Desenvolvimento de Software Ágil Global (AGSD)

Para superar a maioria das complicações enfrentadas por equipes distribuídas globalmente, as empresas estão adotando métodos ágeis para seu ambiente distribuído. No entanto, por definição, práticas ágeis foram desenvolvidas para ajudar equipes co-localizadas (VALLON et al., 2017). Curiosamente, muitos estudos demonstram que métodos ágeis podem ser úteis para mitigar problemas de GSD (RAMESH et al., 2006; RAZZAK et al., 2017; HOSSAIN; BABAR; PAIK, 2009). Nessa perspectiva, os métodos ágeis aparecem como uma metodologia viável, uma vez que são cada vez mais utilizados no desenvolvimento de software em todo o mundo (MARINHO et al., 2019).

Há uma infinidade de revisões de literatura, tanto secundárias como terciárias, sobre AGSD. Essas revisões ocorrem tanto no início da primeira década do Ágil ((JALALI; WOHLIN, 2010)) quanto nas mais recentes ((VALLON et al., 2017)). Além disso, são práticas ágeis em geral no GSD (por exemplo, (VALLON et al., 2017)) ou estão em um tópico específico (HOSSAIN; BABAR; PAIK, 2009). Este artigo relata o uso geral de práticas ágeis no contexto mencionado anteriormente e pode ser comparado diretamente com os estudos (VALLON et al., 2017; JALALI; WOHLIN, 2010; HOSSAIN; BABAR; PAIK, 2009).

Jalali e Wohlin (JALALI; WOHLIN, 2010) mostram que o ágil se tornou mais popular em um ambiente distribuído de 2004 em diante. Além disso, os autores identificaram 25 práticas ágeis em ambientes distribuídos, embora não apresentem a descrição direcionada ao objetivo da prática, que deve ser aplicada à prática, como a prática pode ser adotada em um projeto GSD e quem é o responsável por realização da prática.

Vallon (VALLON et al., 2017) continuou o estudo de Jalali e Wohlin seguindo um desenho de estudo semelhante, para analisar e comparar os resultados mais recentes com os anteriores. Porém, de forma semelhante a eles, as práticas ágeis no GSD não foram descritas de forma a aplicá-lo no contexto distribuído globalmente, e também

apenas casos de sucesso estiveram presentes.

Além disso, Hossain et al. (HOSSAIN; BABAR; PAIK, 2009) conduziu um SLR sobre o uso do Scrum no GSD. O estudo resume um conjunto de desafios que surgem do Scrum no GSD e estratégias para lidar com eles. No entanto, os autores consideraram apenas estudos GSD que se aplicavam ao Scrum.

Portanto, é importante argumentar que nenhuma das revisões anteriores apresentou os resultados (práticas de AGSD) em uma forma *o que*, *quem* e *como*, o que, particularmente, acreditamos tornar este relatório de revisão um documento muito útil também para a indústria.

3 Método de Pesquisa

3.1 Revisão Sistemática da Literatura

Essa Revisão Sistemática da Literatura foi retirada do estudo (CAMARA et al., 2020), publicado na SBES 2020 34th Brazilian Symposium on Software Engineering.

Ao realizar este estudo, adotamos uma abordagem sistemática focada para examinar a literatura relevante. Nosso objetivo não foi descobrir todas as práticas registradas, mas selecionar uma coleção suficiente de estudos para permitir a identificação de temas recorrentes.

As diretrizes de revisão sistemática estabelecidas (KITCHENHAM; CHARTERS, 2007) recomendam que um revisor execute as seguintes etapas:

- (i) identificar a necessidade de uma revisão sistemática da literatura;
- (ii) formular perguntas de pesquisa de revisão;
- (iii) realizar uma busca por estudos relevantes;
- (iv) avaliar e registrar a qualidade dos estudos incluídos;
- (v) classificar os dados necessários para responder às perguntas da pesquisa;
- (vi) extrair dados de cada estudo incluído;
- (vii) resumir e sintetizar os resultados do estudo (meta-análise);
- (viii) interpretar resultados para determinar sua aplicabilidade;
- (ix) escreva os resultados do estudo como um relatório.

Assim, foi elaborado um protocolo de pesquisa (ver Apêndice A.1). Procuramos responder à seguinte pergunta de pesquisa: *"Como as práticas ágeis são adotadas nas equipes ágeis de desenvolvimento de software global?"*. Usamos a seguinte cadeia de pesquisa booleana para garantir que capturamos uma grande variedade de papéis:

((agile OR scrum OR “extreme programming” OR “pair programming” OR “lean development” OR “lean software development” OR SAFe OR “Scaled Agile Framework) AND (“global software engineering” OR “global software development” OR “distributed software engineering” OR “distributed software development” OR “GSE” OR “GSD” OR “distributed team” OR “global team” OR “dispersed team” OR “spread team” OR “virtual team” OR “offshore” OR “outsource”))

O motivo de incluir o SAFe foi porque ele é o framework de escalar ágil mais referenciado pelo versionone.

Usamos essa sequência para pesquisar os metadados relacionados a periódicos e anais de conferências nos bancos de dados bibliográficos:

- IEEEXplore;
- ACM Digital Library;
- SpringerLink;
- Scopus
- Wiley

3.1.1 Seleção de Documentos

A pesquisa produziu 3181 referências de 2001 a 2020 (IEEE = 759; ACM = 289; Springer = 380; Scopus = 887; Wiley = 866). O processo de seleção idealizado teve dois componentes: (fase 1) uma seleção inicial de resultados de pesquisa que pudessem satisfazer razoavelmente os critérios de seleção (descritos a seguir) com base na leitura dos títulos e resumos dos artigos; seguida por (fase 2) uma seleção final de acordo com esses critérios da lista de artigos inicialmente selecionada com base na leitura de suas introduções e conclusões.

3.1.1.1 Critérios de inclusão/exclusão

Os critérios a seguir orientaram a seleção de artigos que nos ajudaram a abordar as questões da pesquisa.

Foram incluídos: (i) artigos publicados completos, revisados por pares; (ii) trabalhos diretamente relacionados às questões de pesquisa; (iii) trabalhos que abordem práticas ágeis no GSD e que o estudo esteja disponível nos serviços de bibliotecas da universidade acessíveis aos autores durante o período da pesquisa.

Foram excluídos: (i) textos não publicados em inglês; (ii) conteúdo técnico, por exemplo: editoriais, tutoriais, palestras, white papers, teses, dissertações, relatórios técnicos, livros; (iii) artigos curtos (<= 4 páginas); e (iv) trabalhos que não estejam claramente relacionados às questões de pesquisa.

Antes de aceitar um artigo no conjunto final para revisão, verificamos se não havia replicação. Por exemplo, se um determinado estudo foi publicado em duas revistas diferentes com uma ordem diferente de autores principais, apenas um estudo seria incluído na revisão; este seria geralmente o estudo mais abrangente ou recente. Além disso, verificamos para garantir que não havia duplicação. Por exemplo, no mesmo artigo listado em mais de um banco de dados, apenas um estudo seria incluído na revisão.

Com esses critérios, identificamos 395 artigos duplicados e nenhuma réplica. Depois de excluir resultados duplicados do conjunto de dados, identificamos papéis para inclusão na seleção inicial (fase 1). Desses papéis, 351 foram encaminhados para a fase 2, na qual 265 foram eliminados e um total de 86 estudos de pesquisa A.2 foram considerados para este SLR. Além disso, apresentamos 48 práticas ágeis identificadas por meio do SLR.

A Tabela 1 apresenta a quantidade de estudos extraídos de cada base de dados por meio da string de pesquisa e quantos foram aceitos em cada fase das extrações.

Base de dados	**Seleção**	**Fase 1**	**Fase 2**
ACM	289	45	22
Scopus	887	143	16
Wiley	866	12	2
Springer	380	33	25
IEEE	759	118	21
Total	**3181**	**351**	**86**

Tabela 1 – Quantidade de artigos por base de dados.

3.1.2 Qualidade do estudo

Os critérios de avaliação de qualidade adotados para nosso estudo são baseados em princípios e boas práticas estabelecidas para impulsionar a pesquisa empírica em engenharia de software (DYBA; DINGSOYR; HANSSEN, 2007). Respondemos às perguntas listadas no Protocolo de Pesquisa (Ver <https://bit.ly/3h4s81s>) usando *Sim, Não, Parcialmente*.

- (i) Existe uma definição clara dos objetivos do estudo?;
- (ii) Existe uma definição clara das justificativas do estudo?;

- (iii) Existe uma base teórica sobre os tópicos do estudo?;
- (iv) Existe uma definição clara da questão de pesquisa (RQ) e/ou da hipótese do estudo?;
- (v) Existe uma descrição adequada do contexto em que a pesquisa foi realizada?;
- (vi) Os métodos de coleta de dados são usados e descritos adequadamente?;
- (vii) Existe uma descrição adequada da amostra utilizada e os métodos para identificar e recrutar a amostra?;
- (viii) Existe uma descrição adequada dos métodos usados para analisar os dados e métodos apropriados para garantir que a análise dos dados seja fundamentada nos dados?;
- (ix) Respostas claras ou justificativas sobre RQ/hipótese são fornecidas pelo estudo?;
- (x) Os resultados claramente declarados com resultados credíveis são fornecidos pelo estudo?;
- (xi) As conclusões justificadas são fornecidas pelo estudo?; e
- (xii) A discussão sobre ameaças de validade é fornecida pelo estudo?;

3.1.3 Extração de Dados

Foram examinadas cada publicação selecionada para extrair os seguintes elementos: (i) objetivo do estudo ou questão de pesquisa, (ii) outros resultados relevantes para o estudo e (iii) possíveis temas emergentes das conclusões do estudo.

Assim, os dados foram sintetizados, identificando primeiro as práticas ágeis de cada artigo como aplicá-las à equipe do GSD. Como atribuímos a cada ocorrência o mesmo peso, as frequências apresentadas refletem simplesmente quantos artigos mencionam uma determinada prática; portanto, as frequências refletem a prevalência de um tema e não sua importância potencial.

Classificamos todos os artigos incluídos em uma das facetas do tipo de pesquisa derivadas de Wieringa *et al.* (WIERINGA et al., 2006). Todos os estudos revisados também foram classificados através de facetas de tipo de contribuição derivadas de Petersen *et al.* (PETERSEN et al., 2008).

- **Opinião:** Não descreve novos resultados de pesquisa. Esses estudos contêm declarações de opinião do autor não fundamentadas em dados empíricos, trabalhos relacionados ou métodos de pesquisa;

- **Solução:** Proposta de solução para um problema, pode ser uma melhoria de uma técnica ou a proposta de uma nova técnica. A solução proposta deve ser discutida e, quando possível, testada e validada;
- **Filosófico:** Propõe uma nova maneira de pensar/olhar para as coisas. Pode ser uma nova estrutura conceitual, uma taxonomia e um estudo secundário, temos RSL ou SMS;
- **Avaliação:** Avaliação de um problema na prática, ou avaliações de uma técnica implementada na prática, por exemplo, estudos de caso;
- **Experiência:** Descreve experiências pessoais. Podem ser experiências pessoais dos autores ou relatórios de experiências industriais.
- **Model:** Representação de uma realidade observada através de conceitos;
- **Framework:** Um conjunto de práticas, métodos e recomendações a serem aplicadas relacionadas ao GSD;
- **Diretriz:** Um conjunto de conselhos, práticas recomendadas e fatores de sucesso baseados em evidências empíricas;
- **Lições aprendidas:** Um conjunto de resultados das descobertas e resultados dos estudos de caso;
- **Recomendações:** Um conjunto de recomendações geralmente da opinião do autor, e não fundamentado em evidências empíricas.

Os dados extraídos por meio do formulário web foram copiados para uma planilha para síntese dos dados. Além disso, a síntese de dados é dividida em síntese quantitativa e qualitativa.

3.2 Survey com Especialistas

Afim de coletar dados a respeito do uso de práticas ágeis e frameworks utilizados no contexto de projetos distribuídos, elaboramos um Survey (MOLLÉRI; PETERSEN; MENDES, 2016).

O Survey está disponível em (<https://forms.gle/9hMXyWMbpvL1zap98>), o protodolo é apresentado no Apêndice B. Nosso Survey é do tipo descritivo, isto é, estamos procurando saber as opiniões dos participantes, utilizamos um questionário com perguntas relacionadas aos costumes dos engenheiros em relação a aplicação de metodologias ágeis distribuídas.

Fizemos perguntas simples e diretas, a maioria delas fechadas, além disso, perguntamos apenas questões que os engenheiros vivenciam diariamente. Distribuímos o Survey em e-mails de universidades, grupos estudantis e entre conhecidos, apesar todo esse esforço, só conseguimos obter 25 respostas válidas, esse é um número baixo, porém, foi possível ter um levantamento inicial das práticas ágeis distribuídas utilizadas atualmente.

4 Resultados

4.1 Resultados da Revisão Sistemática

Nossos resultados da SLR em incluíram 86 artigos e 48 práticas de desenvolvimento de software global ágil para o período de janeiro de 2001 a dezembro de 2020 para serem usados para uma visão geral dos estudos primários e análises qualitativas.

A maioria dos artigos utilizou uma abordagem qualitativa em seus estudos (61 artigos), seguida de uma abordagem mista (20 artigos) e um método quantitativo (4 artigos). Um dos artigos não ficou muito claro como definir o tipo de método de pesquisa.

O mapeamento dos tipos de estudos na Figura 1 mostra as facetas do tipo de pesquisa de todos os estudos do SLR ao longo de 2004-2020. Percebe-se que no início da década de 2000 não foram encontrados estudos sobre práticas ágeis em GSD. Porém, de 2004 a 2011, o número de estudos começou a aumentar, a maioria deles provenientes de experiências, avaliações e soluções. Posteriormente, no período de 2012-2020, podemos observar o surgimento de mais estudos, especificamente estudos filosóficos (13 artigos) que indicam certa maturidade do campo de pesquisa. Embora continuem a ser relatados trabalhos de experiência (17 estudos), mostrando que o campo de pesquisa continua recebendo atenção da academia que pesquisa regularmente na área. Além disso, os tipos de publicação mais comuns ao longo de 2004-2020 foram artigos de avaliação (29 artigos), relatórios de experiência (25 artigos) e estudos filosóficos (19 artigos).

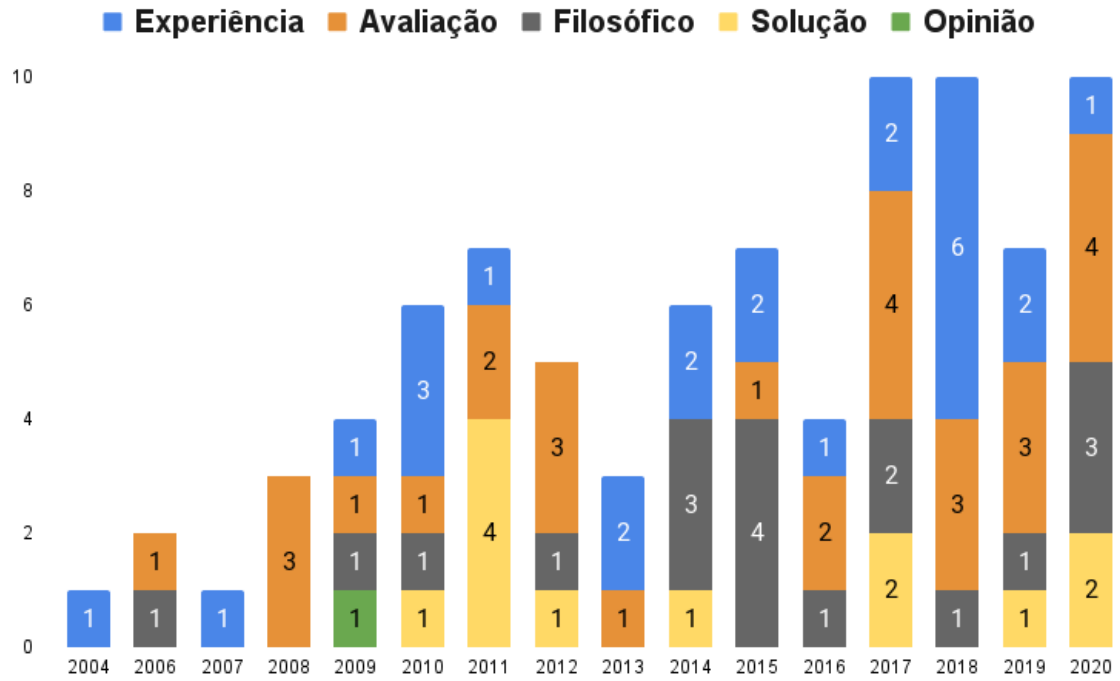

Figura 1 – Facetas de tipo de pesquisa ao longo do tempo.

A distribuição das facetas do tipo de contribuição dos estudos revisados é apresentada na Figura 2. Como podemos ver, os tipos de contribuição mais comuns foram as diretrizes (31 artigos), depois as lições aprendidas (26 artigos), seguidas de recomendações (13 artigos), estruturas (11) artigos e modelo (5 artigos).

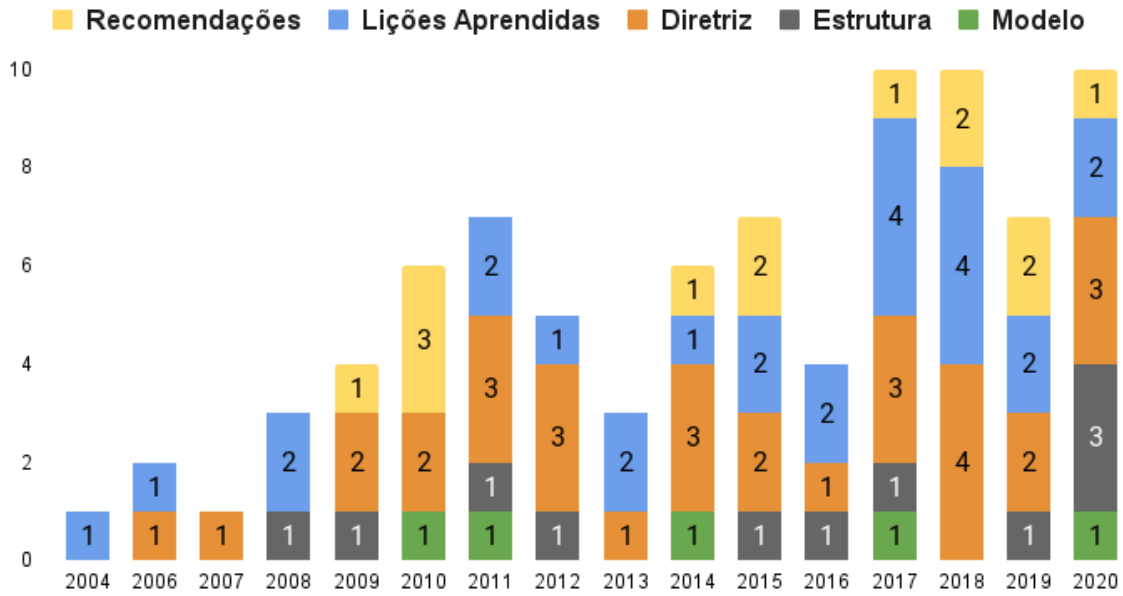

Figura 2 – Facetas de tipo de contribuição ao longo do tempo.

	Ruim (<20%)	Justo (20%-40%)	Médio (40%-60%)	Bom (60%-80%)	Excelente (>80%)
Número de estudos	0	1	5	17	63
Porcetagem trabalhos	0	1,2%	5,8%	19,7%	73,2%

Tabela 2 – Avaliação da qualidade.

A distribuição dos métodos por ano é mostrada na Figura 3. Antes havia artigos que utilizavam mais de um método, portanto o número de métodos não corresponde ao número de artigos. Percebe-se que a maioria dos artigos utilizou a abordagem de estudo de caso (42 artigos), entre os quais foi encontrado pelo menos um artigo, com exceção do ano de 2016, seguido de entrevistas (26 artigos). As metodologias menos utilizadas foram o estudo de casos múltiplos e a pesquisa-ação, com um artigo cada em 2010 e 2016, respectivamente.

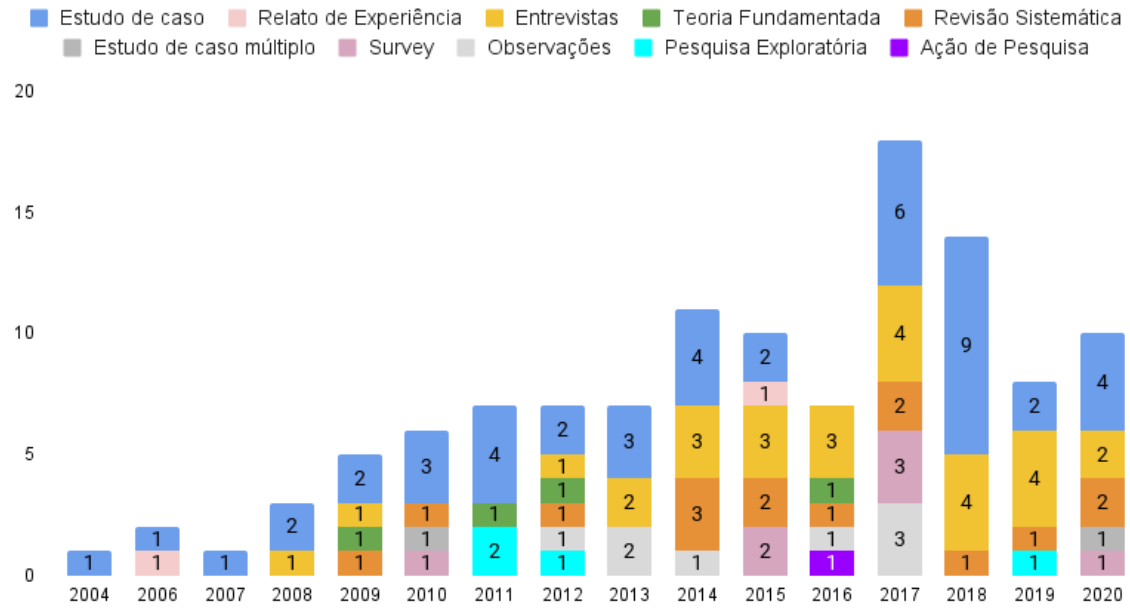

Figura 3 – Distribuição de métodos por ano.

Cada estudo aceito foi avaliado independentemente por quatro pesquisadores, de acordo com doze critérios de qualidade possíveis (ver Seção 3.1.2). Os artigos foram avaliados nas seguintes escalas: <20%, pobre (0 artigos); 20% - 40%, justo (1 artigo); 40% - 60%, média (5 artigos); 60% - 80%, bom (17 artigos); e> 80%, excelente (63 artigos) estes estão listados na Tabela 2. No entanto, a pontuação é apenas uma heurística para ser usada como um guia onde nenhum estudo é rejeitado com base em sua pontuação de qualidade. Para uma comparação justa entre os estudos, normalizamos os dados registrando a pontuação percentual.

4.1.1 RQ: como as práticas ágeis são adotadas nas equipes globais de desenvolvimento de software ágil?

Para ajudar a implementar nosso conjunto de práticas identificadas, iremos descrever cada prática, nas próximas subseções, com base nos estudo de revisão. A descrição visa o objetivo da prática, que deve ser aplicado à prática, quem deve aplicá-la (papéis ágeis) e como a prática pode ser adotada em um ambiente GSD. Nas seções seguintes, o número indicado após cada título de prática refere-se à frequência de aparecimento.

4.1.1.1 Reunião diária (36)

Descrição: A cada dia do Sprint, a equipe realiza um encontro diário, chamada Daily Scrum.

Objetivo: Visa disseminar conhecimento na equipe sobre o que foi feito no dia anterior, identificar impedimentos e informar o trabalho a ser realizado no dia que se inicia.

Quem: Times e scrum master.

Como: No que diz respeito ao GSD, os estudos mencionaram que os encontros foram realizados por telefone, videoconferência, webcam, e-mails, chats na internet entre outros meios de comunicação (ROBINSON, 2019; RAZZAK et al., 2018; LOUS et al., 2018b; SZABÓ; STEGHÖFER, 2019; LOUS; KUHRMANN; TELL, 2017; HOSSAIN; BANNERMAN; JEFFERY, 2011b; LOUS et al., 2018a; ROOPA; KUMAR; MANI, 2018; LAL; CLEAR, 2018; BJØRN; SØDERBERG; KRISHNA, 2019; VALLON et al., 2017; Khmelevsky; Li; Madnick, 2017; KAUSAR; AL-YASIRI, 2017; RAZZAK et al., 2017; MODI; ABBOTT; COUNSELL, 2017; GUPTA; MANIKREDDY, 2015; MOE et al., 2015; RIZVI; BAGHERI; GASEVIC, 2015; ESTáCIO; PRIKLADNICKI, 2014; HUBER; DIBBERN, 2014; SUNDARARAJAN; BHASI; VIJAYARAGHAVAN, 2014; VALLON et al., 2013; RALPH; SHPORTUN, 2013; SRIRAM; MATHEW, 2012; DORAIRAJ; NOBLE; MALIK, 2012; HOSSAIN; BANNERMAN; JEFFERY, 2011a; HILLEGERSBERG; LIGTENBERG; AYDIN, 2011; PAASIVAARA; LASSENIUS, 2010; HOSSAIN; BABAR; VERNER, 2009a; PAASIVAARA; DURASIEWICZ; LASSENIUS, 2008; HOLE; MOE, 2008; KUSSMAUL; JACK; SPONSLER, 2004; BASS, 2015; LEE; YONG, 2009; JALALI; WOHLIN, 2010; HOSSAIN; BABAR; PAIK, 2009).

4.1.1.2 Práticas de comunicação (34)

Descrição: O princípio ágil de indivíduos e interações sobre processos e ferramentas é aquele que reforça a importância da comunicação práticas entre equipes distribuídas ágeis. Uma vez que essas equipes não estão co-localizadas, a comunicação

entre os membros pode ser afetada e, consequentemente, impactar o projeto. Além disso, as diferenças culturais são um grande fator de falha em projetos distribuídos, e para superar isso, as práticas de comunicação são vitais. Muitos artigos apresentaram projetos que discutiam o uso de práticas e políticas de comunicação entre os membros da equipe.

Objetivo: Pretende reduzir mal-entendidos, estreitar os laços das equipes distribuídas e compartilhar informações comuns entre todos os locais.

Quem: Equipes.

Como: Alguns projetos GSD usaram a prática de ”modos de comunicação múltiplos” que combinam uma variedade de abordagens síncronas e assíncronas para estabelecer uma estratégia de comunicação no projeto (LOUS; KUHRMANN; TELL, 2017; HILLEGERSBERG; LIGTENBERG; AYDIN, 2011; HOSSAIN; BABAR; VERNER, 2009b). Além disso, alguns artigos sugerem que as equipes distribuídas devem sincronizar as horas de trabalho para alcançar bons níveis de comunicação (ROBINSON, 2019; RAZZAK et al., 2018; SZABÓ; STEGHÖFER, 2019; HOSSAIN; BANNERMAN; JEFFERY, 2011b; RAMESH et al., 2006; RICHTER; RAITH; WEBER, 2016; HOSSAIN, 2019; VALLON et al., 2017; KAUSAR; AL-YASIRI, 2017; LAUREN, 2015; BANIJAMALI et al., 2017; GUPTA; MANIKREDDY, 2015; MOE et al., 2015; RIZVI; BAGHERI; GASEVIC, 2015; HUBER; DIBBERN, 2014; RAZAVI; AHMAD, 2014; HAMID, 2013; DORAIRAJ; NOBLE; MALIK, 2012; DUMITRIU et al., 2011; HOSSAIN; BANNERMAN; JEFFERY, 2011a; PAASIVAARA; LASSENIUS, 2010; MARUPING, 2010; AVRITZER; BRONSARD; MATOS, 2010; HOSSAIN; BABAR; VERNER, 2009a; PAASIVAARA; DURASIEWICZ; LASSENIUS, 2008; HOLE; MOE, 2008; CHO, 2007; KUSSMAUL; JACK; SPONSLER, 2004; BASS, 2015; HOSSAIN; BABAR; PAIK, 2009; NUEVO; PIATTINI; PINO, 2011).

4.1.1.3 Planning (24)

Descrição: É uma prática na qual os desenvolvedores se reúnem para pensar nas atividades necessárias para atingir um objetivo desejado, como reduzir a distância sociocultural e geográfica, esclarecer mal-entendidos, melhorar a coordenação e a comunicação.

Objetivo: Nas reuniões de planejamento, os desenvolvedores têm a oportunidade de fazer perguntas sobre as tarefas propostas, reduzir mal-entendidos e alinhar o escopo de acordo com as expectativas e feedback do cliente.

Quem: Times de desenvolvimento, PO, scrum master, stakeholders.

Como: O planejamento global de software é aplicado com ferramentas como Skype, ligações e videoconferências, e-mails e visitas (SZABÓ; STEGHÖFER, 2019;

HOSSAIN; BABAR; VERNER, 2009a; PAASIVAARA; DURASIEWICZ; LASSENIUS, 2008). Além disso, de acordo com Rizvi (RIZVI; BAGHERI; GASEVIC, 2015), esses planos ocorrem em ciclos curtos, para reduzir riscos e aumentar o feedback de clientes e outras equipes (LOUS; KUHRMANN; TELL, 2017; HOSSAIN; BANNERMAN; JEFFERY, 2011b; LAL; CLEAR, 2018; VALLON et al., 2017; Khmelevsky; Li; Madnick, 2017; KAUSAR; AL-YASIRI, 2017; BANIJAMALI et al., 2017; RAZZAK et al., 2017; MOE et al., 2015; BRITTO; MENDES; BÖRSTLER, 2015; TRIPATHI et al., 2015; BRITTO; USMAN; MENDES, 2014; SUNDARARAJAN; BHASI; VIJAYARAGHAVAN, 2014; VALLON et al., 2013; TANNER; CHIGONA, 2012; HOSSAIN; BANNERMAN; JEFFERY, 2011a; PAASIVAARA; LASSENIUS, 2010; AVRITZER; BRONSARD; MATOS, 2010; LEE; YONG, 2009; JALALI; WOHLIN, 2010).

4.1.1.4 Scrum dos scrums - SoS (21)

Descrição: Scrum dos Scrums é uma prática usada em equipes muito grandes, nas quais os desenvolvedores são divididos em várias equipes menores. Dessa forma, os scrums de scrums objetivam, por meio do scrum master de cada equipe, realizar reuniões para compartilhar informações e manter todos os membros atualizados sobre os eventos do produto.

Objetivo: O propósito do scrum de scrums é manter as equipes atualizadas sobre os eventos do projeto.

Quem: Scrum master e times.

Como: No contexto do GSD, para aplicar o SoS, as equipes são formadas com base em sua localização, com um membro representante (Scrum master) em cada equipe para garantir a comunicação entre as equipes (KAUSAR; AL-YASIRI, 2017; HOSSAIN; BABAR; PAIK, 2009). Segundo Passivaara e Lassenius (PAASIVAARA; LASSENIUS, 2010), observou-se que as reuniões do SoS podem ser semelhantes às reuniões do scrum com a necessidade de organizar as reuniões virtualmente. Além disso, na reunião de scrum de scrums, o progresso do projeto, decisões de recursos, priorização, abordagens de teste, infraestrutura de codificação compartilhada e gerenciamento de pessoas são discutidos (LOUS; KUHRMANN; TELL, 2017; HOSSAIN; BANNERMAN; JEFFERY, 2011b; ROOPA; KUMAR; MANI, 2018; GUPTA; JAIN; SINGH, 2018; VALLON et al., 2017; Khmelevsky; Li; Madnick, 2017; GUPTA; MANIKREDDY, 2015; RIZVI; BAGHERI; GASEVIC, 2015; SUNDARARAJAN; BHASI; VIJAYARAGHAVAN, 2014; VALLON et al., 2013; LI; MäDCHE, 2013; DUMITRIU et al., 2011; HOSSAIN; BANNERMAN; JEFFERY, 2011a; AVRITZER; BRONSARD; MATOS, 2010; PAASIVAARA; DURASIEWICZ; LASSENIUS, 2008; LEE; YONG, 2009; DINGSøYR et al., 2017; NUEVO; PIATTINI; PINO, 2011).

4.1.1.5 Visitas entre locais (20)

Descrição: As visitas regulares entre os locais é uma prática onde os membros da equipe se reúnem para reduzir a distância sociocultural, melhorar a comunicação e conseguir uma melhor colaboração entre as equipes.

Objetivo: Reduzir a distância sociocultural, melhorar a comunicação e alcançar uma melhor colaboração entre as equipes.

Quem: Equipes e partes interessadas.

Como: Quando acessível, as visitas entre locais são dedicadas a encontros presenciais (GUPTA; MANIKREDDY, 2015). Essas reuniões geralmente acontecem com todos os membros da equipe ou representantes, para que possam discutir questões sobre o projeto, diminuir o distanciamento sociocultural, melhorar a colaboração e a comunicação da equipe. Essas visitas ajudam a construir relacionamentos e confiança (RAJPAL, 2018; HOSSAIN; BABAR; PAIK, 2009; NUEVO; PIATTINI; PINO, 2011), e, de acordo com Szabó e Steghofer, (SZABÓ; STEGHÖFER, 2019) permitem identificar e resolver problemas de forma colaborativa (LOUS; KUHRMANN; TELL, 2017; PAASIVAARA, 2017; HOSSAIN, 2019; MOE et al., 2015; VALLON et al., 2017; RIZVI; BAGHERI; GASEVIC, 2015; VALLON et al., 2013; RALPH; SHPORTUN, 2013; DORAIRAJ; NOBLE; MALIK, 2012; PAASIVAARA; LASSENIUS, 2010; AVRITZER; BRONSARD; MATOS, 2010; HOSSAIN; BABAR; VERNER, 2009b; PAASIVAARA; DURASIEWICZ; LASSENIUS, 2008; HOLE; MOE, 2008; LEE; YONG, 2009).

4.1.1.6 Reunião retrospectiva (19)

Descrição: A Retrospectiva da Sprint ocorre após a Revisão da Sprint e antes do próximo Planejamento da Sprint. Esta é uma reunião máxima de três horas para Sprints de um mês.

Objetivo: Realiza-se uma reunião retrospectiva para verificar e melhorar os processos de execução do projeto.

Quem: Equipes de desenvolvimento e scrum master.

Como: No GSD, as reuniões retrospectivas são realizadas por meio de telefones, videoconferências, webcam e outras ferramentas de comunicação (ROBINSON, 2019; MOE et al., 2015). Algumas das retrospectivas foram realizadas utilizando ferramentas de comunicação síncrona (SZABÓ; STEGHÖFER, 2019), enquanto outras, conforme mencionado nos estudos (KAUSAR; AL-YASIRI, 2017; HOSSAIN; BABAR; PAIK, 2009), foram realizadas de forma assíncrona, através da publicação de comentários e resultados de Wiki e informações compartilhadas por email (LOUS; KUHRMANN; TELL, 2017; HOSSAIN; BANNERMAN; JEFFERY, 2011b; ROOPA; KUMAR; MANI, 2018; VALLON et al., 2017; TRIPATHI et al., 2015; RIZVI; BAGHERI; GASEVIC,

2015; SUNDARARAJAN; BHASI; VIJAYARAGHAVAN, 2014; DORAIRAJ; NOBLE; MALIK, 2012; HOSSAIN; BANNERMAN; JEFFERY, 2011a; PAASIVAARA; LASSENIUS, 2010; HOSSAIN; BABAR; VERNER, 2009a; PAASIVAARA; DURASIEWICZ; LASSENIUS, 2008; DINGSøYR et al., 2017; JALALI; WOHLIN, 2010).

4.1.1.7 Sprint (19)

Descrição: Um sprint é um evento com limite de tempo que pode durar de uma a quatro semanas e, ao final da interação, um incremento de software é produzido e deve ser entregue. Algumas características de um sprint são que mudanças nos requisitos não são permitidas, os objetivos precisam ser definidos e o escopo deve ser negociado entre a equipe e o PO.

Objetivo: Desenvolver requisitos pré-definidos ao longo de um período timeboxed, não aceitando mudanças durante o período e produzindo um incremento de software ao final.

Quem: Equipes.

Como: O sprint é implementado através da aplicação de práticas mais ágeis, como planejamento de sprint, revisão de sprint, retrospectiva de sprint (HOSSAIN; BANNERMAN; JEFFERY, 2011a). Além disso, alguns artigos sugerem manter ciclos de sprints curtos de tamanho para entregar constantemente valor ao cliente (SZABÓ; STEGHÖFER, 2019; LOUS; KUHRMANN; TELL, 2017; HOSSAIN; BANNERMAN; JEFFERY, 2011b; LAL; CLEAR, 2018; ALSAQAF; DANEVA; WIERINGA, 2017; VALLON et al., 2017; GUPTA; MANIKREDDY, 2015; RIZVI; BAGHERI; GASEVIC, 2015; ESTáCIO; PRIKLADNICKI, 2014; RALPH; SHPORTUN, 2013; SRIRAM; MATHEW, 2012; RAMESH; MOHAN; CAO, 2012; HILLEGERSBERG; LIGTENBERG; AYDIN, 2011; PAASIVAARA; LASSENIUS, 2010; AVRITZER; BRONSARD; MATOS, 2010; PAASIVAARA; DURASIEWICZ; LASSENIUS, 2008; HOLE; MOE, 2008; JALALI; WOHLIN, 2010).

4.1.1.8 Product backlog (18)

Descrição: O backlog do produto é uma lista que contém todos os requisitos necessários para desenvolver a solução. O PO é responsável por manter, incluir e priorizar os requisitos no backlog. O product backlog precisa ser constantemente refinado pela equipe e pelo PO, os itens são revisados e refinados por eles.

Objetivo: Visa concentrar todos os recursos de negócios e arquitetônicos que precisam ser implementados por todas as equipes distribuídas.

Quem: PO, líder da equipe e equipe.

Como: De acordo com o estudo de Hossain (HOSSAIN; BANNERMAN; JEFFERY, 2011b), em um dos projetos estudados, a carteira de produtos era refinada e in-

crementada quinzenalmente com o cliente. Lee e Yong (LEE; YONG, 2009) mostraram que as equipes distribuídas estudadas tinham seus próprios backlogs e foram separados de acordo com personalizações locais e regionais (SZABÓ; STEGHÖFER, 2019; LOUS; KUHRMANN; TELL, 2017; GUPTA; JAIN; SINGH, 2018; LAL; CLEAR, 2018; RICHTER; RAITH; WEBER, 2016; ALSAQAF; DANEVA; WIERINGA, 2017; VALLON et al., 2017; Khmelevsky; Li; Madnick, 2017; KAUSAR; AL-YASIRI, 2017; GUPTA; MANIKREDDY, 2015; MOE et al., 2015; RALPH; SHPORTUN, 2013; LI; MäDCHE, 2013; HILLEGERSBERG; LIGTENBERG; AYDIN, 2011; PAASIVAARA; LASSENIUS, 2010; NUEVO; PIATTINI; PINO, 2011).

4.1.1.9 Gerenciamento de backlog (16)

Descrição: O gerenciamento de backlog é a técnica utilizada pelas equipes ágeis para registrar, acompanhar e priorizar o que precisa ser implementado no projeto, para que as metas sejam atendidas.

Objetivo: Garantir a organização das tarefas a serem realizadas.

Quem: PO e equipes.

Como: É importante garantir que os membros da equipe sempre tenham uma visão clara do produto (MOE et al., 2015; LEE; YONG, 2009). Desta forma, os desenvolvedores, de acordo com Paasivaara, Durasiewicz e Lassenius (PAASIVAARA; DURASIEWICZ; LASSENIUS, 2008) usaram uma ferramenta de suporte chamada Jira [1] para gerenciar o backlog e disponibilizá-lo para qualquer pessoa em as equipes distribuídas (ROOPA; KUMAR; MANI, 2018; LAL; CLEAR, 2018; RAMESH et al., 2006; HAMID, 2013; VALLON et al., 2013; LI; MäDCHE, 2013; HOSSAIN; BANNERMAN; JEFFERY, 2011a; HILLEGERSBERG; LIGTENBERG; AYDIN, 2011; MARUPING, 2010; AVRITZER; BRONSARD; MATOS, 2010; HOLE; MOE, 2008; DINGSøYR et al., 2017; JALALI; WOHLIN, 2010).

4.1.1.10 Histórias de usuários (15)

Descrição: As histórias de usuários são práticas que descrevem, de forma detalhada, as necessidades do produto de acordo com a visão do usuário, representando assim um requisito do sistema.

Objetivo: Garantir a compreensão dos requisitos funcionais e não funcionais, fornecendo a visão do usuário final aos membros da equipe.

Quem: PO e equipes.

Como: De acordo com Alsaqaf, Daneva e Wieringa (ALSAQAF; DANEVA; WIERINGA, 2017) no contexto global, as histórias de usuários podem ser distribuídas

[1] www.atlassian.com/software/jira

de acordo com a natureza da história e as habilidades disponíveis nas equipes distribuídas. Outros estudos registraram user stories em plataformas web, como o Jira, disponibilizando-os entre todos os membros da equipe e modificados conforme as necessidades surgiram, garantindo assim um trabalho coordenado entre os desenvolvedores (SZABÓ; STEGHÖFER, 2019; LOUS; KUHRMANN; TELL, 2017; ROOPA; KUMAR; MANI, 2018; LAL; CLEAR, 2018; RAMESH et al., 2006; VALLON et al., 2017; MODI; ABBOTT; COUNSELL, 2017; TRIPATHI et al., 2015; RIZVI; BAGHERI; GASEVIC, 2015; SUNDARARAJAN; BHASI; VIJAYARAGHAVAN, 2014; TANNER; CHIGONA, 2012; HILLEGERSBERG; LIGTENBERG; AYDIN, 2011; JALALI; WOHLIN, 2010; HOSSAIN; BABAR; PAIK, 2009).

4.1.1.11 Programação em pares (15)

Descrição: A técnica de programação em pares é originada do XP, é aplicada emparelhados dois desenvolvedores no desenvolvimento do mesmo código, visando reduzir os bugs e melhorar a qualidade do código devido ao fato de que quando um desenvolvedor está codificando, o outro pode revisar o código.

Objetivo: Desenvolver o código fonte de um projeto através de dois desenvolvedores trabalhando no mesmo computador, visando melhorar a qualidade do código e compartilhar conhecimento.

Quem: Equipes de desenvolvimento.

Como: De acordo com alguns estudos (SCHENK; PRECHELT; SALINGER, 2014; RAZZAK et al., 2017; ESTáCIO; PRIKLADNICKI, 2014) é importante selecionar ferramentas de comunicação adequadas que permitam o compartilhamento de áudio e vídeo para a aplicação de programação distribuída de pares (DPP) (SZABÓ; STEGHÖFER, 2019; LOUS et al., 2018a; LAL; CLEAR, 2018; HOSSAIN, 2019; VALLON et al., 2017; KAUSAR; AL-YASIRI, 2017; GUPTA; MANIKREDDY, 2015; TRIPATHI et al., 2015; RIZVI; BAGHERI; GASEVIC, 2015; MARUPING, 2010; AVRITZER; BRONSARD; MATOS, 2010; JALALI; WOHLIN, 2010).

4.1.1.12 Sprint review (14)

Descrição: A revisão do sprint é uma prática ágil onde o incremento de software desenvolvido no sprint é apresentado pela equipe e posteriormente inspecionado e avaliado pelo PO e stakeholders do projeto. Também é possível para o PO e o cliente, com base nos resultados e feedbacks fornecidos, adaptar o backlog do produto.

Objetivo: Apresentar o incremento do software ao PO e aos stakeholders do projeto com o objetivo de receber feedback sobre os recursos desenvolvidos.

Quem: Equipes, PO, scrum master e stakeholders.

Como: As reuniões de revisão de sprint são geralmente conduzidas por vídeo ou conferências por telefone (KAUSAR; AL-YASIRI, 2017). No entanto, os estudos revisados diferem muito de quem deve estar presente na reunião de revisão, sugerindo apenas a presença do PO (SZABÓ; STEGHÖFER, 2019), ou a presença do mestre scrum, PO e stakeholders do projeto (LOUS; KUHRMANN; TELL, 2017; HOSSAIN; BANNERMAN; JEFFERY, 2011b; VALLON et al., 2017; Khmelevsky; Li; Madnick, 2017; MOE et al., 2015; RIZVI; BAGHERI; GASEVIC, 2015; VALLON et al., 2013; TANNER; CHIGONA, 2012; HOSSAIN; BANNERMAN; JEFFERY, 2011a; HILLEGERSBERG; LIGTENBERG; AYDIN, 2011; HOSSAIN; BABAR; VERNER, 2009a; LEE; YONG, 2009).

4.1.1.13 Autogerenciamento (13)

Descrição: A autogestão é uma abordagem alternativa à gestão. Afasta-se da estrutura hierárquica de gestão tradicional e transfere para os colaboradores as principais atividades de gestão e atividades relacionadas com o trabalho, eliminando efetivamente a função de gestor.

Objetivo: Alinhar prioridades, planos, manter a equipe mais focada no trabalho de forma responsável. É sair da zona de conforto, para conseguir resultados diferentes e mais aprimorados. Além disso, gerar a capacidade de ter empatia com as outras pessoas e controlar as emoções para que elas não interfiram de forma negativa na rotina.

Quem: Equipes.

Como: Diversos estudos (RAMESH et al., 2006; RIZVI; BAGHERI; GASEVIC, 2015; SUNDARARAJAN; BHASI; VIJAYARAGHAVAN, 2014) argumentaram que a autogestão deve ser aplicada permitindo que todos escolham suas tarefas, com foco em atividades de aprendizagem contínua, compartilhamento de conhecimento e formação de equipes, capacidade de compartilhar e definir problemas e atividades quando possível e atribuir responsabilidades exclusivas (GUPTA; JAIN; SINGH, 2018; LAL; CLEAR, 2018; ALSAQAF; DANEVA; WIERINGA, 2017; GERVIGNY; NAGOWAH, 2017; RAZZAK et al., 2017; RAZZAK, 2016; GUPTA; MANIKREDDY, 2015; MOE et al., 2015; TRIPATHI et al., 2015; HUBER; DIBBERN, 2014).

4.1.1.14 Integração contínua (11)

Descrição: As integrações contínuas são uma prática comum de DevOps e tem como objetivo construir constantemente incrementos de soluções cada vez mais valiosos. De acordo com a SAFe, a abordagem de integração contínua deve desenvolver, testar, integrar e validar recursos desenvolvidos em um ambiente de teste, para então, quando estiver pronto, implante e libere-o em um ambiente de produção.

Objetivo: Visa garantir que o valor está sendo entregue ao cliente constantemente por meio de um processo automatizado.

Quem: Equipes de desenvolvimento.

Como: Um estudo relatou que os módulos de solução foram integrados durante a noite para identificar falhas de compilação (SUNDARARAJAN; BHASI; VIJAYARAGHAVAN, 2014). Além disso, o uso de testes de regressão opcionais e relatórios de falha pode ser útil para identificar qualquer problema que possa vir de uma falha na construção (SZABÓ; STEGHÖFER, 2019; LOUS; KUHRMANN; TELL, 2017; VALLON et al., 2017; RIZVI; BAGHERI; GASEVIC, 2015; SRIRAM; MATHEW, 2012; AVRITZER; BRONSARD; MATOS, 2010; HOLE; MOE, 2008; LEE; YONG, 2009; JALALI; WOHLIN, 2010; NUEVO; PIATTINI; PINO, 2011).

4.1.1.15 Projetar a equipe (11)

Descrição: Projetar a equipe é uma prática relacionada às estratégias relacionadas à formação da equipe, à distribuição pelos diferentes locais e à distribuição de membros com as competências adequadas a cada equipe. Alguns artigos sugerem dispersar a maioria dos membros técnicos em diferentes equipes distribuídas e formar equipes com membros localizados em diferentes fusos horários para aprimorar o desenvolvimento na rotina 24/7.

Objetivo: Formar equipes multifuncionais ágeis e distribuídas capazes de alcançar o sucesso do projeto com seus membros.

Quem: PO e scrum master.

Como: Um estudo mostrou um projeto distribuído que tinha uma restrição para distribuição da equipe, no projeto havia vários locais distribuídos, mas uma equipe distribuída ágil só poderia ter membros entre dois locais diferentes (HOSSAIN; BABAR; PAIK, 2009). Além disso, o estudo de Vallon (VALLON et al., 2017) apresentou um projeto distribuído com três equipes scrum formadas em todos os produtos e com base nas áreas de requisitos lógicos (HOSSAIN; BANNERMAN; JEFFERY, 2011b; GUPTA; JAIN; SINGH, 2018; VALLON et al., 2013; LI; MäDCHE, 2013; HILLEGERSBERG; LIGTENBERG; AYDIN, 2011; MARUPING, 2010; HOSSAIN; BABAR; VERNER, 2009b; JALALI; WOHLIN, 2010; NUEVO; PIATTINI; PINO, 2011).

4.1.1.16 Gráficos burndown (11)

Descrição: O gráfico burndown é comumente usado para mostrar a quantidade estimada de atividades pendentes restantes em um período. Por exemplo, sprint burndown é o mais comum e mostra a quantidade estimada de backlog do sprint restante ao longo do tempo de um sprint.

Objetivo: Descobrir a capacidade da equipe, mostrando aos integrantes seu progresso e acurácia de suas estimativas.

Quem: Scrum master.

Como: De acordo com os estudos (RICHTER; RAITH; WEBER, 2016; KAUSAR; AL-YASIRI, 2017; LEE; YONG, 2009), a melhor forma de implementar burndown charts em equipes distribuídas é utilizando ferramentas eletrônicas. Ele disponibiliza os gráficos para qualquer membro em todo o mundo e ajuda a sincronizar as equipes (HOSSAIN; BANNERMAN; JEFFERY, 2011b; VALLON et al., 2017; VALLON et al., 2013; RALPH; SHPORTUN, 2013; TANNER; CHIGONA, 2012; HILLEGERSBERG; LIGTENBERG; AYDIN, 2011; JALALI; WOHLIN, 2010; NUEVO; PIATTINI; PINO, 2011).

4.1.1.17 Sincronizar horário de trabalho (10)

Descrição: A sincronização de horários de trabalho é uma prática utilizada no desenvolvimento ágil distribuído em que equipes que possuem fusos horários distintos buscam ter, pelo menos, algum cruzamento no horário de trabalho.

Objetivo: Aumentar a comunicação e colaboração entre os membros da equipe e reduzir mal-entendidos com horários de trabalho sincronizados.

Quem: Equipes.

Como: É necessário encontrar horários comuns, para que as equipes possam ter comunicação síncrona (SUNDARARAJAN; BHASI; VIJAYARAGHAVAN, 2014; HOSSAIN; BANNERMAN; JEFFERY, 2011a; BASS, 2015). Além disso, segundo Nuevo (NUEVO; PIATTINI; PINO, 2011) os horários das reuniões podem ser alternados, como fazer reuniões no horário normal de trabalho de uma equipe e em outro horário conduzir as reuniões no horário da outra equipe (HILLEGERSBERG; LIGTENBERG; AYDIN, 2011; HOSSAIN; BABAR; VERNER, 2009b; PAASIVAARA; DURASIEWICZ; LASSENIUS, 2008; KUSSMAUL; JACK; SPONSLER, 2004; LEE; YONG, 2009; HOSSAIN; BABAR; PAIK, 2009).

4.1.1.18 Coaching (9)

Descrição: Os estudos relacionados a métodos ágeis de coaching focaram no ensino de métodos ágeis para o cliente, treinando as equipes globalmente distribuídas para o processo ágil, tendo reuniões para reforçar o valor de scrum (HOSSAIN; BABAR; PAIK, 2009), e treinar os membros da equipe para a aplicação de práticas adaptadas como programação distribuída.

Objetivo: Visa treinar as equipes distribuídas, e o cliente em métodos ágeis para alinhá-los com o processo ágil. Além disso, oferece treinamento para os membros da equipe em habilidades técnicas.

Quem: Scrum master e líder da equipe.

Como: De acordo com Rizvi (RIZVI; BAGHERI; GASEVIC, 2015), o scrum master era responsável por treinar a organização para a forma ágil de trabalhar. Além disso, algumas empresas fornecem treinamento em métodos ágeis para as equipes distribuídas globalmente (LEE; YONG, 2009) e outras enviam seu pessoal mais experiente para outro local para treinar os membros juniores (ROOPA; KUMAR; MANI, 2018; MOE et al., 2015; ESTáCIO; PRIKLADNICKI, 2014; SUNDARARAJAN; BHASI; VIJAYARAGHAVAN, 2014; DUMITRIU et al., 2011; HOSSAIN; BABAR; VERNER, 2009b; HOSSAIN; BABAR; PAIK, 2009).

4.1.1.19 Documentação necessária (9)

Descrição: A documentação necessária é a prática de produzir a quantidade mínima de documento no desenvolvimento ágil de software que gere valor.

Objetivo: Melhorar a comunicação e disponibilizar informações sobre o projeto para todos.

Quem: Equipes.

Como: De acordo com o estudo de Hossain (HOSSAIN; BABAR; PAIK, 2009), o uso de plataformas web, como Jira, torna mais fácil compartilhar as informações entre equipes distribuídas online e rastrear os problemas documentado. Além disso, foi observado no estudo (RIZVI; BAGHERI; GASEVIC, 2015) que, ao criar documentação, os membros da equipe reduziram a necessidade constante de comunicação, o que funciona como um benefício, uma vez que a comunicação entre equipes de diferentes culturas pode causar problemas (RICHTER; RAITH; WEBER, 2016; SUNDARARAJAN; BHASI; VIJAYARAGHAVAN, 2014; HAMID, 2013; AVRITZER; BRONSARD; MATOS, 2010; HOSSAIN; BABAR; VERNER, 2009b; PAASIVAARA; DURASIEWICZ; LASSENIUS, 2008; NUEVO; PIATTINI; PINO, 2011).

4.1.1.20 Quadro Kanban (9)

Descrição: O quadro kanban é usado em equipes distribuídas ágeis com o mesmo propósito de equipes co-localizadas, para gerenciar as tarefas a fazer, tarefas em andamento e tarefas feitas.

Objetivo: Promover a visualização e os limites do trabalho em processo em equipes distribuídas em diferentes locais.

Quem: Times, PO, scrum master.

Como: De acordo com os estudos revisados, cada equipe distribuída pode ter seu quadro kanban físico em seus locais (TRIPATHI et al., 2015; VALLON et al., 2013;

DINGSøYR et al., 2017), mas pode dificultar a replicação do status entre os diferentes locais. Além disso, é possível utilizar um quadro eletrônico para todas as equipes distribuídas, o que facilita para qualquer equipe a visualização das atualizações de status das tarefas (HUBER; DIBBERN, 2014; HILLEGERSBERG; LIGTENBERG; AYDIN, 2011).

4.1.1.21 Co-localizar todos os membros da equipe no início (8)

Descrição: Reunião com toda a equipe no início do projeto.

Objetivo: Facilitar a interação da equipe mesmo por um curto período de tempo. Tem como objetivo estabelecer a confiança em toda a equipa.

Quem: Equipes.

Como: De acordo com os estudos revisados (AVRITZER; BRONSARD; MATOS, 2010; HOSSAIN; BABAR; PAIK, 2009; NUEVO; PIATTINI; PINO, 2011), os projetos ágeis distribuídos optam por reunir todos os membros da equipe, quando possível, no início do projeto.

4.1.1.22 Gerenciamento de tarefas (8)

Descrição: O gerenciamento de tarefas é uma prática que gerencia uma tarefa em todo o seu processo (pronta, atribuída, concluída, expirada, encaminhada, iniciada, concluída, verificada, pausada, falhada) para ajudar os desenvolvedores a colaborar e compartilhar conhecimento sobre o projeto.

Objetivo: Ajudar na colaboração e coordenação da equipe durante a execução das tarefas do projeto.

Quem: Times, PO e scrum master.

Como: Kausar e Yasiri encontraram no estudo (KAUSAR; AL-YASIRI, 2017) que a prática é usada por meio de ferramentas de compartilhamento online, nas quais toda a equipe pode ter acesso ao backlog do produto, storyboard, taskboard, burndown graphics e outros artefatos ágeis. Além disso, um repositório central de código foi usado para que as equipes possam ver o andamento das tarefas (HOSSAIN; BANNERMAN; JEFFERY, 2011b; LOUS et al., 2018a; BANIJAMALI et al., 2017; MODI; ABBOTT; COUNSELL, 2017; CHO, 2007; KUSSMAUL; JACK; SPONSLER, 2004; HOSSAIN; BABAR; PAIK, 2009).

4.1.1.23 Wiki do projeto (7)

Descrição: As informações sobre projetos distribuídos precisam estar disponíveis para qualquer pessoa em diferentes locais, por isso, uma prática importante se-

guida pelo agile distribuído equipes é a construção de um projeto wiki.

Objetivo: Concentrar toda a informação referente ao projeto distribuído em um só lugar que possa ser acessível a qualquer membro da equipe.

Quem: Equipes.

Como: O wiki do projeto é normalmente usado para armazenar informações sobre a documentação do sistema, testes funcionais, diretrizes arquitetônicas, rotinas de equipe e retrospectivas de sprint (DINGSøYR et al., 2017). Além disso, estudos mostraram que equipes distribuídas ágeis geralmente têm um wiki online colaborativo para armazenar a documentação do projeto (AVRITZER; BRONSARD; MATOS, 2010; HOSSAIN; BABAR; VERNER, 2009a; HOSSAIN; BABAR; VERNER, 2009b; LEE; YONG, 2009; HOSSAIN; BABAR; PAIK, 2009; NUEVO; PIATTINI; PINO, 2011).

4.1.1.24 Implantação Contínua (6)

Descrição: A implantação contínua é uma prática de desenvolvimento de software na qual as alterações incrementais feitas no código são testadas, controladas e implantadas automaticamente no ambiente de desenvolvimento, para que as atualizações possam chegar aos clientes o mais rápido possível.

Objetivo: Garantir que os recursos em estágios sejam constantemente implantados no servidor de produção por meio de um processo automatizado de testes de integração e implantação.

Quem: Equipes de desenvolvimento.

Como: No contexto ágil, a implantação contínua é utilizada, (AVRITZER; BRONSARD; MATOS, 2010) por meio de atualizações contínuas na base de código. Assim, quando um código defeituoso é adicionado à base do projeto, uma notificação é enviada a todos os desenvolvedores que contribuíram com esta atualização, garantindo assim que as falhas foram detectadas em tempo hábil no projeto.

4.1.1.25 Colaboração entre equipes (6)

Descrição: As equipes de negócios devem colaborar com as equipes técnicas (equipes ágeis) para fornecer infraestrutura, contratos e fornecedores, treinamento do usuário final, orientação jurídica, marketing, conhecimento da solução e etc.

Objetivo: Visa aumentar a colaboração da equipe, gerando confiança e criando um ambiente agradável.

Quem: Equipes.

Como: Segundo alguns estudos, as sessões de colaboração podem ser feitas antes de uma reunião oficial, para esclarecer dúvidas, compartilhar problemas, ou ape-

nas socializar (PAASIVAARA, 2017; DORAIRAJ; NOBLE; MALIK, 2012; TANNER; CHIGONA, 2012; DORAIRAJ; NOBLE; MALIK, 2011; DINGSøYR et al., 2017; HOSSAIN; BABAR; PAIK, 2009).

4.1.1.26 Entrega contínua (6)

Descrição: Entrega contínua é o que chamamos de uma série de práticas projetadas para garantir que o código possa ser implantado com rapidez e segurança na produção, fornecendo todas as alterações em um ambiente de trabalho e garantindo que os aplicativos e serviços de negócios funcionem conforme o esperado por meio de testes automatizados rigorosos.

Objetivo: Entregar incrementos de software ao cliente com freqüência.

Quem: Equipes de desenvolvimento.

Como: Os estudos revisados mencionaram as entregas de código como uma atividade frequente cujo cronograma pode ser definido com base nas necessidades do projeto (PAASIVAARA; DURASIEWICZ; LASSENIUS, 2008; KUSSMAUL; JACK; SPONSLER, 2004). Porém, um estudo sugere a entrega de valores desde as fases iniciais do projeto (RAJPAL, 2018; LAL; CLEAR, 2018; RAMESH et al., 2006; HOLE; MOE, 2008).

4.1.1.27 Reunião de estimativa (6)

Descrição: É uma reunião onde os membros da equipe conversam entre si e decidem em quanto tempo cada tarefa poderá ser concluída.

Objetivo: Medir o esforço necessário para desenvolver as tarefas do projeto e cumprir os prazos de entrega. Além de realizar estimativas utilizando o tamanho de um software.

Quem: Equipes de desenvolvimento.

Como: Os estudos revisados não descrevem adequadamente como conduzir reuniões de estimativa. No entanto, como muitas das reuniões distribuídas, precisa ser por meio de plataformas baseadas na web, como videoconferência, audioconferências. No entanto, os estudos descreveram o uso de muitas técnicas de estimativa, como tamanho da tarefa, pontos da história, pontos de caso de uso, etc (VALLON et al., 2017; GONÇALVES et al., 2017; BRITTO; MENDES; BÖRSTLER, 2015; TRIPATHI et al., 2015; BRITTO; USMAN; MENDES, 2014; SUNDARARAJAN; BHASI; VIJAYARAGHAVAN, 2014).

4.1.1.28 Demo System (6)

Descrição: O Demo System é um evento significativo que fornece uma visão integrada de novos Recursos para a Iteração mais recente entregue por todas as equipes no Agile Release Train (ART).

Objetivo: Demonstrar os requisitos desenvolvidos na última iteração. Tem como objetivo receber feedback sobre o trabalho realizado.

Quem: PO, equipes de desenvolvimento, scrum master e partes interessadas.

Como: De acordo com Passivara (PAASIVAARA; DURASIEWICZ; LASSENIUS, 2008), as demonstrações foram organizadas através de teleconferência e compartilhamento de aplicativos, com a participação dos membros da equipe, o PO e o scrum master (LAL; CLEAR, 2018; SUNDARARAJAN; BHASI; VIJAYARAGHAVAN, 2014; DINGSøYR et al., 2017; JALALI; WOHLIN, 2010; HOSSAIN; BABAR; PAIK, 2009).

4.1.1.29 Revisão de código (6)

Descrição: A revisão do código é uma prática de desenvolvimento de software em que um ou mais desenvolvedores verificam partes do código desenvolvido e corrigem os defeitos encontrados, a fim de garantir a qualidade do produto e compartilhar o conhecimento entre os membros da equipe.

Objetivo: Melhorar a qualidade do código, detectar bugs antecipadamente, compartilhar conhecimento, reduzir testes e garantir a qualidade da solução.

Quem: Equipes de desenvolvimento.

Como: Rizvi estudo (RIZVI; BAGHERI; GASEVIC, 2015) apresenta que recursos seniores podem auxiliar na revisão do código devido às suas habilidades. Além disso, um estudo sugere que os desenvolvedores seniores realizem a revisão do código junto com os desenvolvedores juniores (MOE et al., 2015).

4.1.1.30 Desenvolvimento orientado a testes - TDD (6)

Descrição: O desenvolvimento orientado a testes (TDD) é uma prática de desenvolvimento de software em que os testes são escritos antes dos recursos estarem prontos, de forma que o código produzido seja feito para passar nesses testes, a fim de evitar desperdício na codificação.

Objetivo: Garantir qualidade, evitar desperdício de código e melhorar o processo de desenvolvimento.

Quem: Equipes de desenvolvimento.

Como: De acordo com Hossain, Babar e Verner (HOSSAIN; BABAR; VERNER,

2009a), o TDD permitiu uma visão padronizada de desenvolvimento, o que facilitou um melhor entendimento de qual funcionalidade era necessária sob a visão do cliente (HOSSAIN; BABAR; VERNER, 2009a). Além disso, um estudo sugeriu desenvolver apenas o código necessário para executar todos os testes e, se os testes forem bem-sucedidos, a próxima etapa seria refatorar o código (LAL; CLEAR, 2018; VALLON et al., 2017; TRIPATHI et al., 2015; JALALI; WOHLIN, 2010; NUEVO; PIATTINI; PINO, 2011).

4.1.1.31 Automação de teste (6)

Descrição: O teste de automação é uma técnica de teste de software para testar e comparar o resultado real com o resultado esperado.

Objetivo: A automação de teste visa executar muitos casos de teste automática e repetidamente quando uma nova versão do projeto é implantada.

Quem: Equipes de desenvolvimento.

Como: De acordo com Passivara (PAASIVAARA; DURASIEWICZ; LASSENIUS, 2008) estudo de caso, o produto desenvolvido costumava ser construído todas as noites com um conjunto de testes automatizados, e se um construído não teve sucesso na equipe distribuída o responsável por esta construção seria o responsável pelo conserto (LOUS; KUHRMANN; TELL, 2017; VALLON et al., 2017; RAZZAK et al., 2017; HOSSAIN; BABAR; VERNER, 2009b; JALALI; WOHLIN, 2010).

4.1.1.32 Gerenciar as expectativas do cliente (6)

Descrição: Como o nome já diz, é uma forma de gerenciar as expectativas do cliente.

Objetivo: Manter o cliente ciente do que está acontecendo no projeto, mantendo a comunicação e visibilidade do andamento do projeto.

Quem: PO.

Como: É necessário manter contato frequente com o cliente para gerenciar suas expectativas (MOE et al., 2015; RAMESH; MOHAN; CAO, 2012), e torná-lo parte do time (DORAIRAJ; NOBLE; MALIK, 2012). Além disso, compartilhar o conteúdo do backlog e o andamento da solução com o cliente pode ajudar na gestão da expectativa (VALLON et al., 2017; AVRITZER; BRONSARD; MATOS, 2010; JALALI; WOHLIN, 2010).

4.1.1.33 Jogo de planejamento (6)

Descrição: O jogo de planejamento é uma prática de estimativa no desenvolvimento de softwares ágeis feita em interações semanais (pode variar conforme o projeto). Nessas reuniões, os requisitos são priorizados e estimados pelos desenvolvedores com o cliente, a fim de obter os requisitos priorizados.

Objetivo: Planeje as tarefas priorizadas estimando-as com técnicas de jogo.

Quem: Equipes de desenvolvimento, PO e scrum master.

Como: De acordo com o estudo de Maruping (MARUPING, 2010), é mais eficaz realizar o jogo do plano por meio de comunicação síncrona, como videoconferência ou telefone. Além disso, a técnica de planning poker apareceu como uma técnica comum para o planejamento de jogos em equipes distribuídas (VALLON et al., 2017; KAUSAR; AL-YASIRI, 2017; RIZVI; BAGHERI; GASEVIC, 2015; BRITTO; USMAN; MENDES, 2014; JALALI; WOHLIN, 2010).

4.1.1.34 Atribuir um papel a cada membro do projeto (5)

Descrição: Em (GUPTA; MANIKREDDY, 2015), os autores descreveram funções relacionadas ao escalonamento ágil, como proprietário do produto chefe (CPO), mestre do scrum principal (CSM), proprietário do produto parcial (PPO), etc.

Objetivo: Atribuir uma função a cada membro do projeto das equipes distribuídas com base em suas habilidades existentes.

Quem: Líder da equipe.

Como: As funções da equipe de uma equipe de desenvolvimento ágil distribuída devem ser atribuídas aos membros do projeto com base em suas habilidades e conhecimentos existentes. Como visto em (GUPTA; MANIKREDDY, 2015) o foco deve ser dado para formar uma equipe multifuncional com todo o conhecimento necessário para desenvolver a solução (SUNDARARAJAN; BHASI; VIJAYARAGHAVAN, 2014; VALLON et al., 2013; RALPH; SHPORTUN, 2013; LI; MäDCHE, 2013).

4.1.1.35 Arquitetura ágil (5)

Descrição: A arquitetura ágil é a aplicação do pensamento enxuto por meio do relacionamento entre a estrutura organizacional de seus desenvolvedores, visando agregar mais valor ao cliente por meio da eliminação de desperdícios.

Objetivo: Garantir a simplicidade do desenho da arquitetura da solução enquanto o processo ágil é conduzido.

Quem: Líder de equipe e arquitetos.

Como: Arquitetura ágil pode ser usada para remover ineficiências do processo de desenvolvimento de software (RIZVI; BAGHERI; GASEVIC, 2015). As reuniões de arquitetura ágil são comumente conduzidas por meio de reuniões de videoconferência entre arquitetos e líderes de equipe para discutir melhorias em toda a arquitetura e seus componentes, para agregar valor ao produto (HILLEGERSBERG; LIGTENBERG; AYDIN, 2011; JHA; VILARDELL; NARAYAN, 2016). Além disso, em outro estudo (HOSSAIN; BABAR; PAIK, 2009), os autores relatam que uma arquitetura independente com interfaces bem definidas permite que as equipes sejam mais autônomas (DINGSøYR et al., 2017).

4.1.1.36 Padrões de codificação (5)

Descrição: Os padrões de codificação são um conjunto formalizado de regras e práticas às quais os desenvolvedores podem aderir ao escrever o código, o que garante que o código seja facilmente legível, mantido e extensível e reduz o risco de introdução de bugs.

Objetivo: Auxiliar os desenvolvedores na formalização de padrões comuns para escrever código legível e sustentável.

Quem: Líder de equipe e equipes de desenvolvimento.

Como: foi sugerido que os padrões de codificação precisam ser discutidos e estabelecidos no início do processo de desenvolvimento (MARUPING, 2010). E os membros seniores do projeto devem disseminar os padrões e garantir que todos os membros da equipe tenham entendido as regras que nortearão o desenvolvimento. Além disso, os padrões de codificação precisam ser seguidos por todos os locais distribuídos (VALLON et al., 2017; HOSSAIN; BABAR; VERNER, 2009a; LEE; YONG, 2009; JALALI; WOHLIN, 2010).

4.1.1.37 Refatoração (5)

Descrição: Refatoração é uma técnica focada em reestruturar um trecho de código existente, alterando sua estrutura interna sem mudar seu comportamento externo.

Objetivo: Simplificar a manutenção, melhorar a qualidade e compreensão do código. *Quem*: Equipes de desenvolvimento.

Como: De acordo com o estudo de caso de Szabó e Steghofer (SZABÓ; STEGHÖFER, 2019), as sessões de refatoração foram conduzidas uma ou duas vezes por mês em seu projeto distribuído (VALLON et al., 2017; HOSSAIN; BABAR; VERNER, 2009a; LEE; YONG, 2009; JALALI; WOHLIN, 2010).

4.1.1.38 Propriedade do código coletivo (4)

Descrição: A propriedade coletiva do código é uma prática que diz que qualquer desenvolvedor da equipe pode alterar qualquer linha de código, adicionar novos recursos, resolver bugs, melhorar o design ou refatorar. O objetivo é encorajar os membros da equipe de que eles são responsáveis pelo design do sistema.

Objetivo: Aplicar propriedade coletiva do código para encorajar cada membro da equipe a ser responsável pelo design do sistema. Além disso, seja capaz de alterar qualquer linha de código, adicionar recursos, corrigir bugs e refatorar.

Quem: Equipes de desenvolvimento.

Como: De acordo com Maruping (MARUPING, 2010), os membros da equipe precisam estar cientes de seus papéis e responsabilidades antes da implementação da propriedade coletiva do código e para isso é necessária uma reunião envolvendo todos os locais (SZABÓ; STEGHÖFER, 2019; TRIPATHI et al., 2015; LEE; YONG, 2009).

4.1.1.39 Feedback frequente (4)

Descrição: O feedback frequente é a prática de dar feedback às pessoas sobre o seu trabalho de acordo com as expectativas de uma tarefa específica.

Objetivo: Melhorar a comunicação entre os membros da equipe e o cliente, motivar os membros da equipe e reduzir as falhas de comunicação.

Quem: Líder da equipe e partes interessadas.

Como: As sessões de feedback para a equipe podem ser feitas mensalmente para entender como cada membro pode melhorar para atingir os objetivos (GUPTA; MANIKREDDY, 2015). Além disso, podem ser feitos loops de feedback com o cliente e a equipe para evitar problemas nos requisitos desenvolvidos (RIZVI; BAGHERI; GASEVIC, 2015; MOE et al., 2015; AVRITZER; BRONSARD; MATOS, 2010).

4.1.1.40 Rastreamento de bug (4)

Descrição: Um sistema de rastreamento de bugs ou sistema de rastreamento de defeitos é um aplicativo de software que rastreia os bugs de software relatados em projetos de desenvolvimento de software. Pode ser considerado um tipo de sistema de rastreamento de problemas.

Objetivo: Manter o controle de bugs relatados e informações sobre eles.

Quem: Equipes de desenvolvimento.

Como: Vários artigos mencionaram que usavam ferramentas da web para rastrear problemas e erros, como Jira (HOSSAIN; BABAR; VERNER, 2009b; CHO, 2007;

HOSSAIN; BABAR; PAIK, 2009) e Bugzilla (LEE; YONG, 2009). Todos eles, com o mesmo objetivo de rastrear as questões que emergem da solução.

4.1.1.41 Documentação das lições aprendidas (4)

Descrição: Documentar as lições aprendidas é uma prática no desenvolvimento ágil na qual os desenvolvedores se reúnem para marcar acertos e erros a fim de melhorar o trabalho em equipe em iterações futuras.

Objetivo: Melhorar o trabalho em equipe em iterações futuras.

Quem: Equipes.

Como: De acordo com Rizvi, Bagheri e Gasevic (RIZVI; BAGHERI; GASEVIC, 2015), é recomendado documentar as lições aprendidas ao final de cada sprint. Além disso, o uso do repositório de lições aprendidas foi relatado (SUNDARARAJAN; BHASI; VIJAYARAGHAVAN, 2014; PAASIVAARA; DURASIEWICZ; LASSENIUS, 2008; HOSSAIN; BABAR; PAIK, 2009).

4.1.1.42 Planejamento do roteiro (4)

Descrição: O planejamento do roteiro é uma prática de desenvolvimento ágil que busca flexibilizar o planejamento das principais etapas do projeto para atingir os objetivos do negócio.

Objetivo: Ajudar a equipe a entender a visão do projeto.

Quem: PO e líder da equipe.

Como: Segundo Hossain (HOSSAIN; BABAR; PAIK, 2009), constatou-se que o planejamento do roteiro ocorreu em reuniões trimestrais, contribuindo para um melhor entendimento da visão do projeto. Além disso, de acordo com Lal e Tony (LAL; CLEAR, 2018), o uso do planejamento de roadmap permitiu à equipe tomar decisões mais coletivas, expandiu a preocupação com o tempo de lançamento no mercado e melhorou a identificação de requisitos de alto nível para projetos curtos (TANNER; CHIGONA, 2012; NUEVO; PIATTINI; PINO, 2011).

4.1.1.43 Compartilhamento da missão e visão (3)

Descrição: Compartilhar uma missão e visão comum é uma prática popular no SAFe, e devido ao número de equipes distribuídas em um software global projeto de desenvolvimento, tal prática foi vista algumas vezes na revisão. O objetivo da prática é alinhar todos os membros da equipe global com o objetivo do solução compartilhando a missão e a visão do projeto

Objetivo: Alinhar todas as equipes distribuídas a um objetivo, visão e missão comuns para o desenvolvimento da solução com base em requisitos viáveis e valiosos para os usuários.

Quem: PO e scrum master.

Como: De acordo com os estudos revisados, a prática de compartilhar missão e visão pode ser aplicada conduzindo a equipe a um processo de desenvolvimento baseado em requisitos viáveis, justificados e priorizados, sempre considerando a experiência do usuário final (LAL; CLEAR, 2018).

4.1.1.44 Rotação de membros da equipe entre locais (3)

Descrição:Esta prática diz que as pessoas podem deixar sua equipe por uma equipe diferente, ou por uma empresa completamente diferente.

Objetivo: Alternar os membros da equipe entre diferentes locais para melhorar as relações ponto a ponto e construir confiança.

Quem: Líder da equipe e mestre do scrum.

Como: Foi relatado em um estudo que alguns membros da equipe alternam sua localização próxima à localização do cliente (DORAIRAJ; NOBLE; MALIK, 2012). Além disso, um estudo de caso sugeriu a rotação de membros entre as equipes, especificamente quando novos membros chegaram e as equipes existentes precisaram ser divididas (PAASIVAARA; DURASIEWICZ; LASSENIUS, 2008; DINGSøYR et al., 2017).

4.1.1.45 Design simples (3)

Descrição: Uma equipe que adota a prática de Design simple, baseia sua estratégia de design de software em atividade contínua, qualidade e padrões.

Objetivo: Manter o design da solução simples ao longo do processo de desenvolvimento.

Quem: Equipes de desenvolvimento.

Como: O estudo (SZABÓ; STEGHÖFER, 2019) relata que o design simples pode ser afetado pela distância sociocultural, levando à tomada de decisões sem as devidas informações técnicas (VALLON et al., 2017; LEE; YONG, 2009).

4.1.1.46 Testes de aceitação (3)

Descrição: O teste de aceitação é um nível de teste de software em que um sistema é testado quanto à aceitabilidade.

Objetivo: Avaliar a conformidade do sistema de acordo com os requisitos do negócio e as necessidades do cliente.

Quem: PO, partes interessadas e equipes.

Como: Os estudos revisados não especificam como realizar os testes de aceitação.

4.1.1.47 Gerenciamento de testes (2)

Descrição: A prática de gerenciamento de teste está relacionada a um plano de teste e estratégia de relatório de teste (NUEVO; PIATTINI; PINO, 2011). O plano de teste é organizado contendo todos os testes que precisam ser realizados, e as ferramentas e recursos necessários para isso (NUEVO; PIATTINI; PINO, 2011). Além disso, o relatório de teste contém os scripts usados no teste, os resultados dos testes realizados e as resoluções dos problemas (JHA; VILARDELL; NARAYAN, 2016).

Objetivo: Gerenciar os testes do projeto através de planos de teste e relatórios de teste.

Quem: Testadores e equipes de desenvolvimento.

Como: De acordo com Madan e Rosa (JHA; VILARDELL; NARAYAN, 2016), o gerenciamento de teste pode ser conduzido por meio de reuniões semanais virtuais entre equipes distribuídas, com o objetivo de discutir o progresso relacionado ao teste, dependências e resolução de problemas (NUEVO; PIATTINI; PINO, 2011).

4.1.1.48 Expandir a responsabilidade da equipe gradualmente (2)

Descrição: A expansão de responsabilidades em equipes ágeis é uma prática no desenvolvimento ágil em que o líder da equipe permite que a equipe faça suas próprias escolhas com base no compartilhamento de informações suficientes.

Objetivo: Tornar os membros da equipe mais responsáveis e capazes de realizar as tarefas do projeto de forma independente.

Quem: Líder da equipe.

Como: No contexto do GSD, a expansão das responsabilidades ocorre por meio de pequenos passos, introduzindo gradativamente os recursos de backlog mais importantes para as equipes (MOE et al., 2015; NUEVO; PIATTINI; PINO, 2011).

4.2 Resultados do Survey

4.2.1 Visão geral dos participantes

A maioria dos participantes possuem experiência com desenvolvimento de software entre: 3 e 5 anos, com 8 respostas. Porém, 6 participantes responderam entre 1 e 2 anos, e outros 6 participantes responderam que tem mais de 10 anos de experiência (Veja 4).

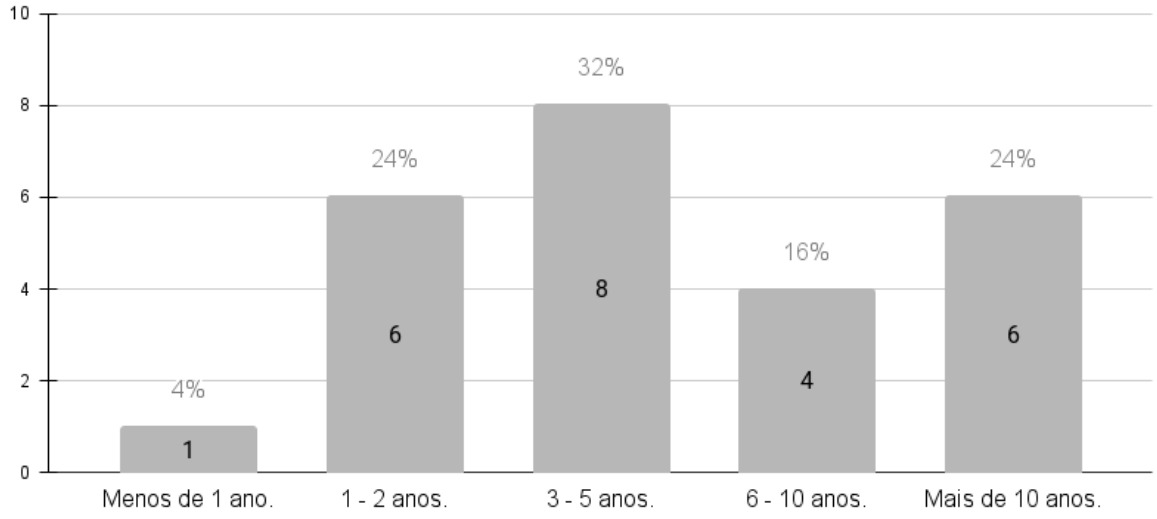

Figura 4 – Anos de experiência com Desenvolvimento de Software

A respeito do tamanho das empresas, e mais da metade dos participantes (13) responderam que trabalham em empresas grandes ou muito grandes, com 251 funcionários ou mais. (Ver Figura 5)

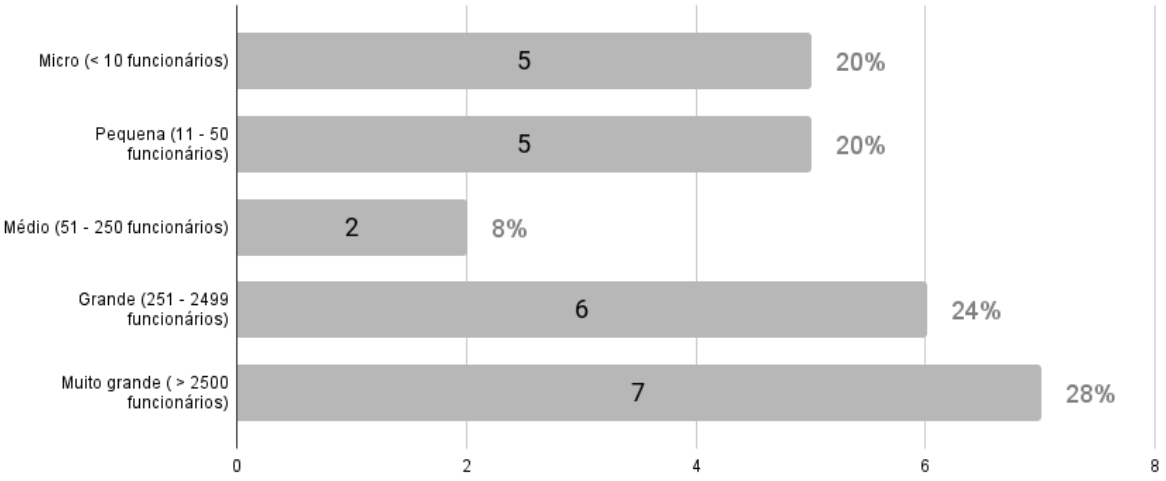

Figura 5 – Tamanho das empresas

Perguntamos aos participantes qual a principal área de negócios da empresas que eles trabalhavam e 10 deles responderam ”Desenvolvimento de Software”, em segundo lugar com 4 respostas ”Desenvolvimento de Sistema”, seguidos por ”Consultoria, treinamento e serviços e TI” e ”Pesquisa e Desenvolvimento”, ambos com 3 de respostas dos participantes (Ver Figura 6).

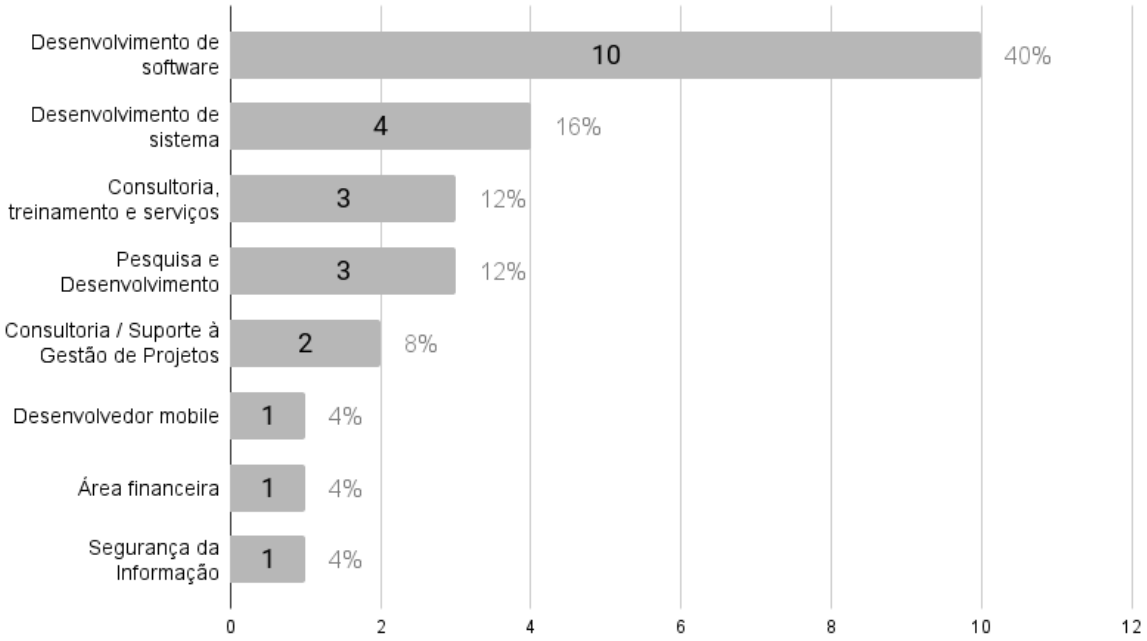

Figura 6 – Principal área de negócios da empresa

Quanto ao modo de atuação das empresas, 7 são empresas que atuam nacionalmente, ou seja, no mesmo país, 2 regionalmente, no mesmo continente, e 7 responderam globalmente, no mundo inteiro.

Além disso, foi perguntado a função que cada um dos participantes desempenha nas suas empresas, e mais da metade (13) respondeu “Desenvolvedor”. Seguidos de “Gerente de Produto”, “Gerente de Projeto”, “Scrum Master” e “Testador”, com 2 respostas cada (Ver Figura 7).

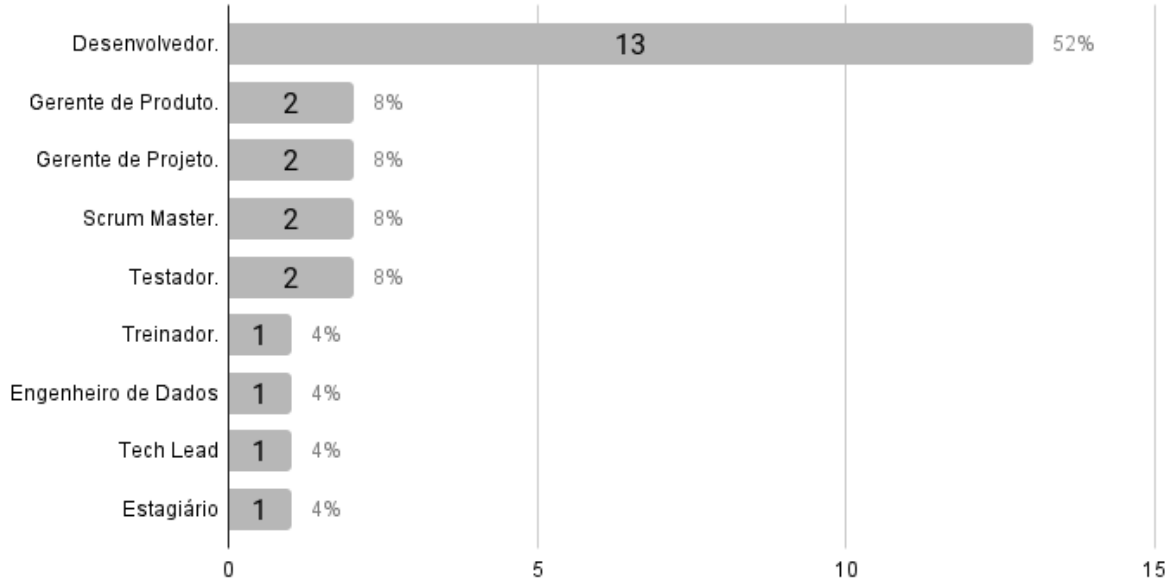

Figura 7 – Função dos participantes

4.2.2 Frameworks e Abordagens de desenvolvimento de software

Pedimos aos participantes para nos informar os frameworks que eles utilizam mais frequentemente. Nesta pergunta, era possível adicionar um framework da escolha do participante.

Dentre os frameworks listados na Figura 8, os 25 participantes afirmaram que o Scrum e Kanban foram os mais utilizados com 20 e 15 respostas, respectivamente. Seguidos de DevOps e Desenvolvimento Interativo, ambos com 7 respostas.

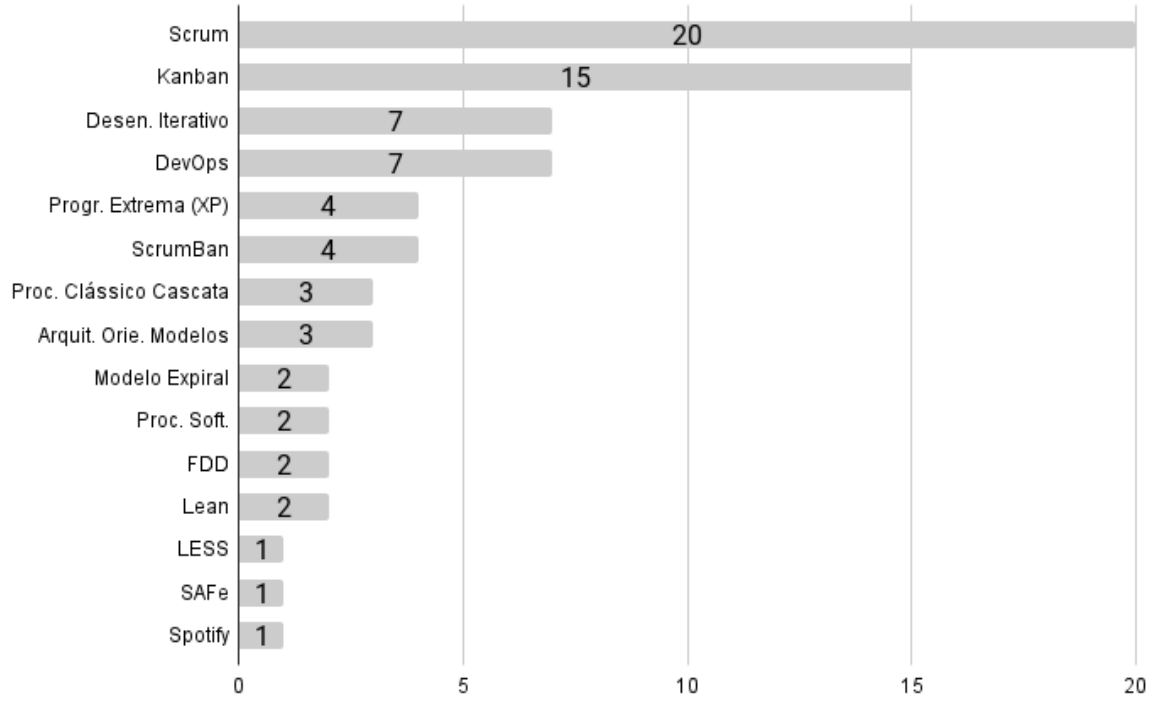

Figura 8 – Respostas dos frameworks mais utilizados

Ainda, as abordagens mais utilizadas de acordo com as 25 respostas dos participantes são principalmente ágeis ou balanceado entre ágeis e tradicionais, totalizando 17 respostas (Ver Figura 9).

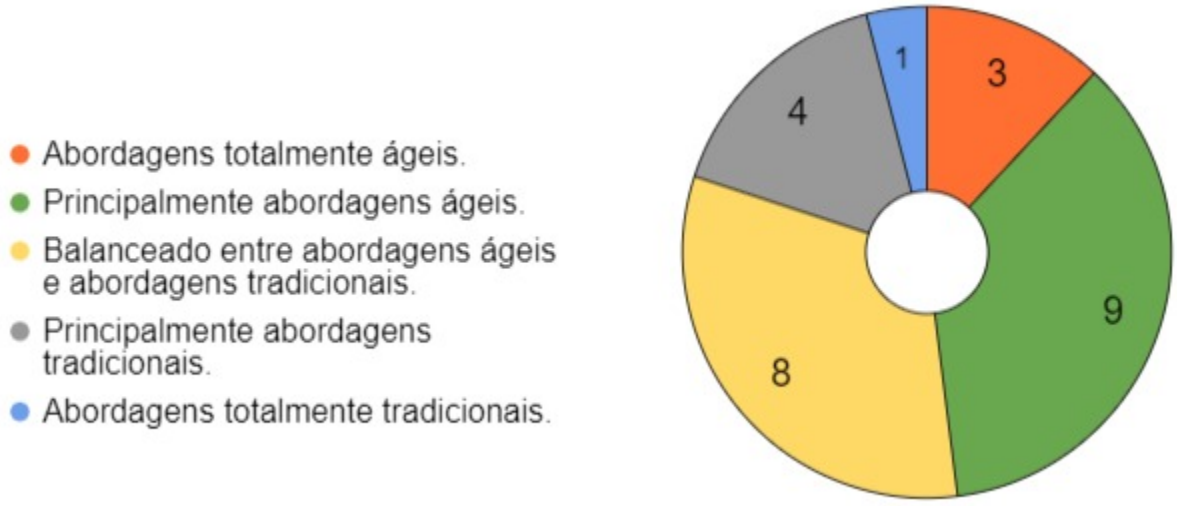

Figura 9 – Abordagens de desenvolvimento de software

4.2.3 Frequência de uso das práticas AGSD

A Figura 10 mostra a frequência de utilização das práticas AGSD's pelos participantes da pesquisa. Cada um teve a oportunidade de responder "Não uso", "Uso raramente", "As vezes uso", "Uso frequentemente" ou "Sempre uso", sobre as 48 práticas ágeis. Como podemos observar na Figura 10, Sprint e Reunião diária, foram as práticas mais utilizadas pelos participantes, com 16 e 14 respostas, respectivamente. Por outro lado, a prática menos utlizada foi "Visita entre locais de trabalho", 14 participantes afirmam nunca terem usado a prática.

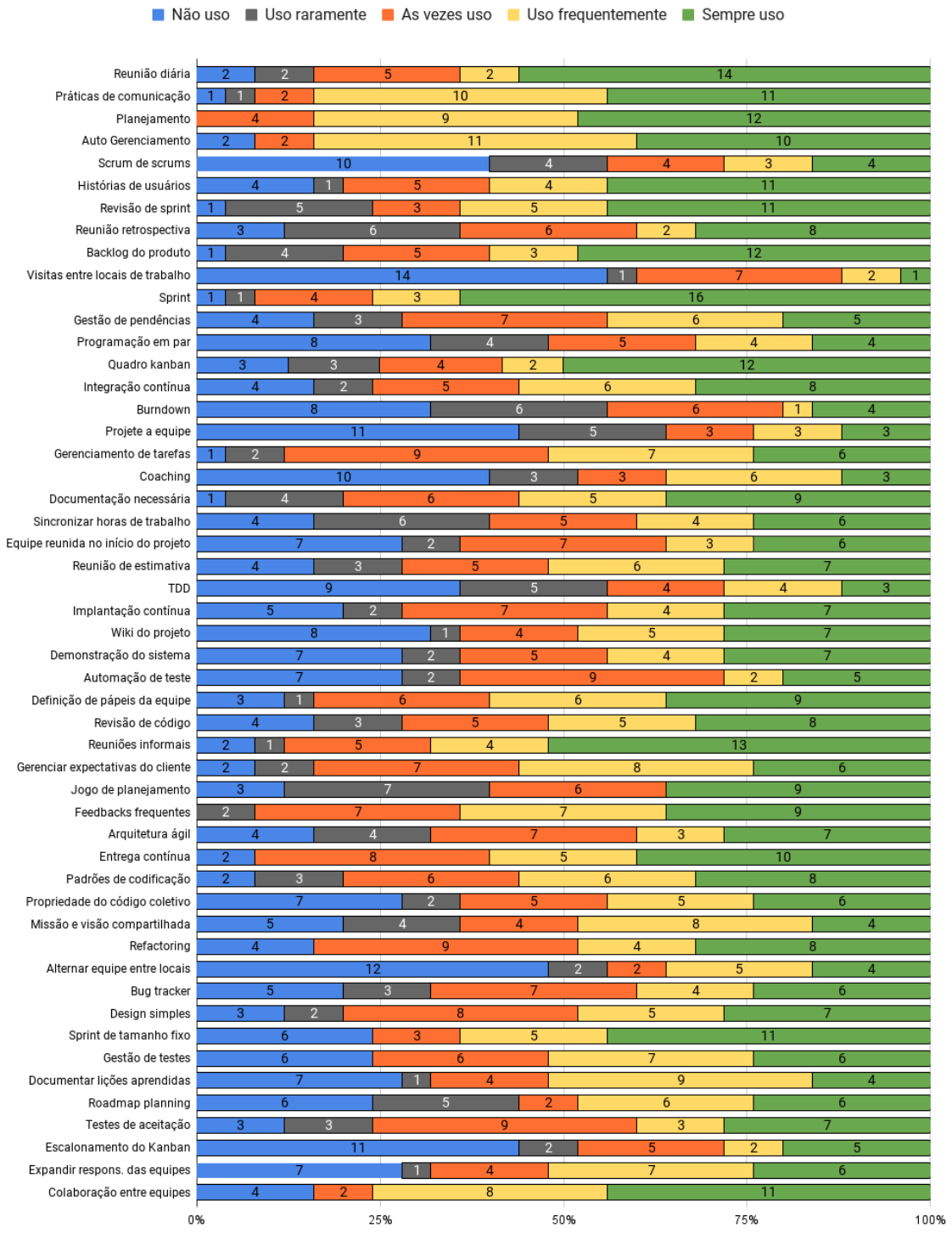

Figura 10 – Utilização de práticas ágeis

5 Discussão

Na revisão sistemática da literatura, encontramos 48 práticas ágeis sendo aplicadas em GSD. Reuniões diárias (ver Seção 4.1.1.1), práticas de comunicação (ver Seção 4.1.1.2) e planejamento (ver Seção 4.1.1.3) foram as práticas mais citadas, respectivamente com 36, 34 e 24 citações. Percebemos que ambas as práticas têm um ponto comum, a comunicação, que enfatiza um valor importante do manifesto ágil: indivíduos e interações sobre processos e ferramentas (BECK et al., 2001). Portanto, podemos reafirmar que esse valor também é seguido na AGSD, pois as reuniões diárias, as práticas de comunicação e o planejamento promovem o controle e a coordenação das equipes globais, agregando valor e compartilhando as informações necessárias para todos os integrantes do projeto.

Apesar dos benefícios do uso de práticas ágeis em GSD, existem alguns desafios em relação à adaptação, controle e coordenação dessas práticas em um ambiente global. Horário de trabalho sincronizado (ver Seção 4.1.1.17) é uma das práticas mais difíceis de serem adotadas, pois é muito difícil sincronizar o horário de trabalho para equipes distribuídas com grandes diferenças de fuso horário. Além disso, coloque todos os membros da equipe no início (consulte a Seção 4.1.1.21), organize as visitas entre os locais (consulte a Seção 4.1.1.5) e faça a rotação dos membros da equipe entre os locais (consulte a Seção 4.1.1.44) não são práticas baratas ou acessíveis para todos os projetos distribuídos, o que torna essas práticas difíceis também.

Além dos desafios enfrentados, é importante apontar para a popularidade das práticas do Scrum no AGSD uma vez que oito das dez práticas mais citadas dos estudos revisados são do Scrum, sendo reuniões diárias (ver Seção 4.1.1.1), planejamento (veja a Seção 4.1.1.3), scrum de scrums (veja a Seção 4.1.1.4), reunião retrospectiva (veja a Seção 4.1.1.6), sprint (veja a Seção 4.1.1.7), product backlog (veja Seção 4.1.1.8), gerenciamento de backlog (veja Seção 4.1.1.9), histórias de usuário (veja Seção 4.1.1.10. É possível dizer que Scrum é o método ágil de maior sucesso aplicado de 2004 a 2020 no GSD. Além disso, os estudos que relataram o uso de tais práticas ágeis mostram como elas são adaptáveis em um ambiente global. Apontando que a maioria das reuniões e eventos apresentados em métodos ágeis pode ser manipulado com o uso de ferramentas online, aplicativos baseados na web, etc (SZABÓ; STEGHÖFER, 2019; RIZVI; BAGHERI; GASEVIC, 2015; HOSSAIN; BABAR; PAIK, 2009).

Observamos que o número de publicações se tornou mais constante a partir de 2011 (ver Figura 3). Além disso, observamos no mesmo gráfico o número crescente de estudos de avaliação de 2014 para cá, aparecendo mais observações, levantamentos

e estudos de entrevistas aliados a estudos de caso que podem fornecer resultados mais substanciais. Além disso, é possível observar o crescimento da SLR (ver Figura 1) que indica a maturidade que a área de pesquisa está ganhando. No entanto, é importante dizer que a academia e a indústria continuam trabalhando juntas relatando suas experiências por meio de estudos de caso. Figura 3 confirma que na última década o número de artigos de estudo de caso continuou a crescer.

Com as 25 respostas que obtivemos do Survey, foi possível observar que, assim como na revisão sistemática, práticas como “Reunião Diária”, “Sprint”, “Planejamento” ficaram entre as mais citadas, no Survey, estas mesmas práticas foram marcadas como as mais utilizadas (ver Figura 10). Assim, é reforçado o que foi apresentado na RSL, o quão estas práticas são importantes tanto no contexto ágil quanto no contexto GSD. Apesar de mais da metade dos participantes da pesquisa serem Desenvolvedores (ver Figura 7), apenas 40% tem mais de 6 anos de experiência com Desenvolvimento de software (ver Figura 4), desta forma, acreditamos que isso influencia diretamente nos resultados da pesquisa e por isso, entendemos que a pesquisa precisa de um número maior de respostas para avaliação dos resultados na prática.

5.1 Limitações

Com relação a RSL, apesar do estudo ter sido realizado por meio de um protocolo de pesquisa estável, que foi discutido e validado por todos os autores para garantir a confiabilidade, ainda apresenta algumas limitações. Apenas cinco bases de dados bibliográficas foram consideradas neste estudo, portanto, é possível que alguns estudos válidos de outras bases de dados não tenham sido incluídos. Porém, para garantir a validade da conclusão, selecionamos as bases de dados mais renomadas para pesquisas ágeis, como ACM, IEEE, Springer, Scopus e Wiley.

Além disso, para reduzir o viés do pesquisador e garantir a validade do construto, quatro pesquisadores foram envolvidos nas fases da pesquisa, desde a concepção até a seleção dos estudos e extração de dados. Se a seleção do estudo em qualquer fase não foi aceita por um dos autores, todos discutiram o artigo para chegar a um acordo sobre a inclusão ou exclusão do estudo. No entanto, replicar o protocolo de pesquisa não garante que outros pesquisadores obtenham os mesmos resultados devido à diferença de decisões individuais, atualizações de bancos de dados e avaliação do processo de inclusão e exclusão. Com relação ao Survey, apesar de dispararmos a pesquisa para inúmeras plataformas, e-mails de universidades e grupos estudantis, a quantidade de dados (25 resposta) ainda foi muito baixa. Esperavamos ter pelo menos 50 respostas para ter uma amostragem maior e desta forma, obter conclusões mais confiáveis.

6 Conclusão

Este trabalho de conclusão contribui para uma coleção de experiências relatadas na literatura sobre como as práticas foram aplicadas aos diferentes ASD em projetos GSD para mitigar desafios com a engenharia de software distribuída. Essas práticas são o resultado de uma síntese sistemática de recomendações encontradas nos estudos publicados.

Em nosso estudo, realizamos uma revisão sistemática detalhada de como as práticas são adotadas em equipes globais. 86 estudos publicados entre 2004 e 2020 foram selecionados. Esses estudos incluem 48 práticas ágeis usadas no desenvolvimento global de software. A literatura tem nos mostrado que essas práticas estão sendo adaptadas ao contexto global, mitigando os desafios do GSD.

Com as 25 respostas que obtivemos, conseguimos observar que o Survey complementou a RSL, mostrando que muitos desenvolvedores de software, seja de grandes empresas ou não, estão trabalhando em locais diferentes e utilizando várias práticas e abordagens ágeis nas suas rotinas. Desta forma, é possível concluir que os dados do Survey confirmou o que foi visto na RSL.

Portanto, como trabalho futuro, pretendemos fazer uma republicação do Survey, afim de coletar mais respostas para ter um levantamento mais conclusivo a respeito da utilização das práticas e frameworks atualmente.

Referências

ALSAQAF, W.; DANEVA, M.; WIERINGA, R. Quality requirements in large-scale distributed agile projects – a systematic literature review. In: *International Working Conference on Requirements Engineering: Foundation for Software Quality*. Cham: Springer, 2017. p. 219–234. ISBN 978-3-319-54045-0. Citado 3 vezes nas páginas 28, 29 e 31.

ALTAF, A. et al. A systematic literature review on factors impacting agile adaptation in global software development. In: *Proceedings of the 2019 7th International Conference on Computer and Communications Management*. New York, NY, USA: ACM, 2019. (ICCCM 2019), p. 158–163. ISBN 9781450371957. Citado na página 10.

AVRITZER, A.; BRONSARD, F.; MATOS, G. Improving global development using agile. In: ____. *Agility Across Time and Space: Implementing Agile Methods in Global Software Projects*. Berlin, Heidelberg: Springer, 2010. p. 133–148. ISBN 978-3-642-12442-6. Citado 12 vezes nas páginas 25, 26, 27, 28, 29, 30, 32, 34, 35, 36, 39 e 42.

BANIJAMALI, A. et al. Empirical investigation of scrumban in global software development. In: *Model-Driven Engineering and Software Development*. Cham: Springer, 2017. p. 229–248. ISBN 978-3-319-66302-9. Citado 3 vezes nas páginas 25, 26 e 35.

BASS, J. How product owner teams scale agile methods to large distributed enterprises. *Empirical Software Engineering*, v. 20, p. 1525–1557, 12 2015. Citado 3 vezes nas páginas 24, 25 e 33.

BECK, K. et al. *Manifesto for Agile Software Development*. 2001. Disponível em: <http://www.agilemanifesto.org/>. Citado 2 vezes nas páginas 10 e 52.

BJØRN, P.; SØDERBERG, A.-M.; KRISHNA, S. Translocality in global software development: the dark side of global agile. *Human–Computer Interaction*, v. 34, p. 174 – 203, 2019. Citado na página 24.

BRITTO, R.; MENDES, E.; BÖRSTLER, J. An empirical investigation on effort estimation in agile global software development. In: *2015 IEEE 10th International Conference on Global Software Engineering*. Ciudad Real, Spain: IEEE, 2015. p. 38–45. Citado 2 vezes nas páginas 26 e 37.

BRITTO, R.; USMAN, M.; MENDES, E. Effort estimation in agile global software development context. In: *Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation*. Cham: Springer, 2014. p. 182–192. ISBN 978-3-319-14358-3. Citado 3 vezes nas páginas 26, 37 e 40.

CAMARA, R. et al. Agile global software development: A systematic literature review. In: *Proceedings of the 34th Brazilian Symposium on Software Engineering*. New York, NY, USA: Association for Computing Machinery, 2020. (SBES '20), p. 31–40. ISBN 9781450387538. Disponível em: <https://doi.org/10.1145/3422392.3422411>. Citado na página 15.

CHO, J. Distributed scrum for large-scale and mission-critical projects. In: *AMCIS*. [S.l.: s.n.], 2007. v. 1, p. 235. Citado 4 vezes nas páginas 25, 35, 42 e 43.

DINGSøYR, T. et al. Exploring software development at the very large-scale: a revelatory case study and research agenda for agile method adaptation. *Empirical Software Engineering*, v. 23, p. 1–31, 06 2017. Citado 12 vezes nas páginas 12, 26, 27, 28, 29, 34, 35, 36, 37, 38, 41 e 44.

DORAIRAJ, S.; NOBLE, J.; MALIK, P. Bridging cultural differences: A grounded theory perspective. In: *Proceedings of the 4th India Software Engineering Conference*. New York, NY, USA: ACM, 2011. (ISEC '11), p. 3–10. ISBN 9781450305594. Citado na página 37.

DORAIRAJ, S.; NOBLE, J.; MALIK, P. Understanding team dynamics in distributed agile software development. In: *Agile Processes in Software Engineering and Extreme Programming*. Berlin, Heidelberg: Springer, 2012. p. 47–61. Citado 7 vezes nas páginas 24, 25, 27, 28, 37, 39 e 44.

DUMITRIU, F. et al. Issues and strategy for agile global software development adoption. *Proceedings of the World Multiconference on Applied Economics, Business and Development*, v. 209, p. 37–42, 01 2011. Citado 3 vezes nas páginas 25, 26 e 34.

DYBA, T.; DINGSOYR, T.; HANSSEN, G. K. Applying systematic reviews to diverse study types: An experience report. In: *1st Int'l Conference on Empirical Software Engineering and Measurement (ESEM) 2007.* Madrid, Spain: IEEE, 2007. p. 225–234. Citado na página 17.

ESTáCIO, B.; PRIKLADNICKI, R. A set of practices for distributed pair programming. In: *ICEIS*. [S.l.: s.n.], 2014. v. 2. Citado 4 vezes nas páginas 24, 28, 30 e 34.

GERVIGNY, M.; NAGOWAH, S. Knowledge sharing for agile distributed teams: A case study of mauritius. In: *2017 International Conference on Infocom Technologies and Unmanned Systems (Trends and Future Directions) (ICTUS)*. Dubai, United Arab Emirates: IEEE, 2017. p. 413–419. Citado na página 31.

GONÇALVES, W. et al. Using agile methods in distributed software development environments. In: *Agile Methods*. Cham: Springer, 2017. p. 16–27. ISBN 978-3-319-55907-0. Citado na página 37.

GUPTA, R. K.; JAIN, S.; SINGH, B. Challenges in scaling up a globally distributed legacy product: A case study of a matrix organization. In: *Proceedings of the 13th International Conference on Global Software Engineering*. New York, NY, USA: ACM, 2018. (ICGSE '18), p. 77–81. ISBN 9781450357173. Citado 4 vezes nas páginas 26, 29, 31 e 32.

GUPTA, R. K.; MANIKREDDY, P. Challenges in adapting scrum in legacy global configurator project. In: *2015 IEEE 10th International Conference on Global Software Engineering*. Ciudad Real, Spain: IEEE, 2015. p. 46–50. Citado 10 vezes nas páginas 24, 25, 26, 27, 28, 29, 30, 31, 40 e 42.

HAMID, A. M. E. Upgrading distributed agile development. In: *2013 International Conference on Computing, Electrical and Electronic Engineering (ICCEEE)*. Khartoum, Sudan: IEEE, 2013. p. 709–714. Citado 3 vezes nas páginas 25, 29 e 34.

HILLEGERSBERG, J. van; LIGTENBERG, G.; AYDIN, M. N. Getting agile methods to work for cordys global software product development. In: *New Studies in Global IT and Business Service Outsourcing*. Berlin, Heidelberg: Springer, 2011. p. 133–152. ISBN 978-3-642-24815-3. Citado 11 vezes nas páginas 10, 24, 25, 28, 29, 30, 31, 32, 33, 35 e 41.

HOLE, S.; MOE, N. B. A case study of coordination in distributed agile software development. In: *Software Process Improvement*. Berlin, Heidelberg: Springer, 2008. p. 189–200. ISBN 978-3-540-85936-9. Citado 7 vezes nas páginas 24, 25, 27, 28, 29, 32 e 37.

HOSSAIN, E.; BABAR, M. A.; PAIK, H.-y. Using scrum in global software development: A systematic literature review. In: *2009 Fourth IEEE International Conference on Global Software Engineering*. Limerick, Ireland: IEEE, 2009. p. 175–184. Citado 19 vezes nas páginas 12, 13, 14, 24, 25, 26, 27, 30, 32, 33, 34, 35, 36, 37, 38, 41, 42, 43 e 52.

HOSSAIN, E.; BABAR, M. A.; VERNER, J. How can agile practices minimize global software development co-ordination risks? In: *Software Process Improvement*. Berlin, Heidelberg: Springer, 2009. p. 81–92. ISBN 978-3-642-04133-4. Citado 9 vezes nas páginas 24, 25, 26, 27, 28, 31, 36, 39 e 41.

HOSSAIN, E.; BABAR, M. A.; VERNER, J. Towards a framework for using agile approaches in global software development. In: *Product-Focused Software Process Improvement*. Berlin, Heidelberg: Springer, 2009. p. 126–140. ISBN 978-3-642-02152-7. Citado 9 vezes nas páginas 25, 27, 32, 33, 34, 36, 39, 42 e 43.

HOSSAIN, E.; BANNERMAN, P. L.; JEFFERY, D. R. Scrum practices in global software development: A research framework. In: *Product-Focused Software Process Improvement*. Berlin, Heidelberg: Springer, 2011. p. 88–102. ISBN 978-3-642-21843-9. Citado 8 vezes nas páginas 24, 25, 26, 27, 28, 29, 31 e 33.

HOSSAIN, E.; BANNERMAN, P. L.; JEFFERY, R. Towards an understanding of tailoring scrum in global software development: A multi-case study. In: *Proceedings of the 2011 International Conference on Software and Systems Process*. New York, NY, USA: ACM, 2011. (ICSSP '11), p. 110–119. ISBN 9781450307307. Citado 9 vezes nas páginas 24, 25, 26, 27, 28, 31, 32, 33 e 35.

HOSSAIN, S. S. Challenges and mitigation strategies in reusing requirements in large-scale distributed agile software development: A survey result. In: *Intelligent Computing-Proceedings of the Computing Conference*. Cham: Springer, 2019. p. 920–935. Citado 3 vezes nas páginas 25, 27 e 30.

HUBER, T.; DIBBERN, J. How collaboration software enables globally distributed software development teams to become agile - an effective use perspective. In: *Governing Sourcing Relationships. A Collection of Studies at the Country, Sector and Firm Level*. Cham: Springer, 2014. p. 49–63. ISBN 978-3-319-11367-8. Citado 4 vezes nas páginas 24, 25, 31 e 35.

JALALI, S.; WOHLIN, C. Agile practices in global software engineering - a systematic map. In: *2010 5th IEEE International Conference on Global Software Engineering*. Princeton, NJ, USA: IEEE, 2010. p. 45–54. Citado 13 vezes nas páginas 13, 24, 26, 27, 28, 29, 30, 32, 33, 38, 39, 40 e 41.

JHA, M. M.; VILARDELL, R. M. F.; NARAYAN, J. Scaling agile scrum software development: Providing agility and quality to platform development by reducing time to market. In: *2016 IEEE 11th International Conference on Global Software Engineering (ICGSE)*. Irvine, CA, USA: IEEE, 2016. p. 84–88. Citado 3 vezes nas páginas 12, 41 e 45.

KAUSAR, M.; AL-YASIRI, A. Using distributed agile patterns for supporting the requirements engineering process. In: ____. *Requirements Engineering for Service and Cloud Computing*. Cham: Springer, 2017. p. 291–316. ISBN 978-3-319-51310-2. Citado 10 vezes nas páginas 24, 25, 26, 27, 29, 30, 31, 33, 35 e 40.

Khmelevsky, Y.; Li, X.; Madnick, S. Khmelevsky, youry and li, xitong and madnick, stuart. In: *2017 Annual IEEE International Systems Conference (SysCon)*. Montreal, QC, Canada: IEEE, 2017. p. 1–4. Citado 4 vezes nas páginas 24, 26, 29 e 31.

KITCHENHAM, B.; CHARTERS, S. *Guidelines for performing systematic literature reviews in software engineering*. [S.l.], 2007. Citado 2 vezes nas páginas 10 e 15.

KUSSMAUL, C.; JACK, R.; SPONSLER, B. Outsourcing and offshoring with agility: A case study. In: *Extreme Programming and Agile Methods - XP/Agile Universe 2004*. Berlin, Heidelberg: Springer, 2004. p. 147–154. ISBN 978-3-540-27777-4. Citado 5 vezes nas páginas 24, 25, 33, 35 e 37.

LAL, R.; CLEAR, T. Enhancing product and service capability through scaling agility in a global software vendor environment. In: *Proceedings of the 13th International Conference on Global Software Engineering*. New York, NY, USA: ACM, 2018. (ICGSE '18), p. 59–68. ISBN 9781450357173. Citado 11 vezes nas páginas 24, 26, 28, 29, 30, 31, 37, 38, 39, 43 e 44.

LAUREN, B. S. Mapping the workspace of a globally distributed "agile" team. *Int. J. Sociotechnology Knowl. Dev.*, IGI Global, USA, v. 7, n. 2, p. 45–62, abr. 2015. ISSN 1941-6253. Citado na página 25.

LEE, S.; YONG, H.-S. Distributed agile: project management in a global environment. *Empirical Software Engineering*, v. 15, p. 204–217, 2009. Citado 13 vezes nas páginas 24, 26, 27, 29, 31, 32, 33, 34, 36, 41, 42, 43 e 44.

LI, Y.; MäDCHE, A. Formulating effective coordination strategies in agile global software development teams. In: *Digital Innovation in the Service Economy. 33th International Conference on Information Systems (ICIS), Orlando, USA, December 16-19, 2012. Vol.: 5*. Orlando, FL, USA: Red Hook, Curran (NY), 2013. p. 4438–4449. ISBN 978-1-62748-604-0. Citado 4 vezes nas páginas 26, 29, 32 e 40.

LOUS, P.; KUHRMANN, M.; TELL, P. Is scrum fit for global software engineering? In: *Proceedings of the 12th International Conference on Global Software Engineering*. Buenos Aires, Argentina: IEEE, 2017. (ICGSE '17), p. 1–10. ISBN 9781538615874. Citado 10 vezes nas páginas 24, 25, 26, 27, 28, 29, 30, 31, 32 e 39.

LOUS, P. et al. From scrum to agile: A journey to tackle the challenges of distributed development in an agile team. In: *Proceedings of the 2018 International Conference on Software and System Process*. New York, NY, USA: ACM, 2018. (ICSSP '18), p. 11–20. ISBN 9781450364591. Citado 3 vezes nas páginas 24, 30 e 35.

LOUS, P. et al. Virtual by design: How a work environment can support agile distributed software development. In: *Proceedings of the 13th International Conference on Global Software Engineering*. New York, NY, USA: ACM, 2018. (ICGSE '18), p. 102–111. ISBN 9781450357173. Citado 2 vezes nas páginas 10 e 24.

MARINHO, M.; NOLL, J.; BEECHAM, S. Uncertainty management for global software development teams. In: *2018 11th International Conference on the Quality of Information and Communications Technology (QUATIC)*. Coimbra, Portugal: IEEE, 2018. p. 238–246. Citado 2 vezes nas páginas 12 e 13.

MARINHO, M. et al. Plan-driven approaches are alive and kicking in agile global software development. In: *International Symposium on Empirical Software Engineering and Measurement (ESEM)*. Porto de Galinhas, Brazil: IEEE, 2019. p. 1–11. Citado na página 13.

MARUPING, L. M. Implementing extreme programming in distributed software project teams: Strategies and challenges. In: ____. *Agility Across Time and Space: Implementing Agile Methods in Global Software Projects*. Berlin, Heidelberg: Springer, 2010. p. 11–30. ISBN 978-3-642-12442-6. Citado 7 vezes nas páginas 25, 29, 30, 32, 40, 41 e 42.

MODI, S.; ABBOTT, P.; COUNSELL, S. Exploring the emergence of collaborative practices in globally distributed agile software development. In: *AMCIS*. Boston, USA: Association for Information Systems, 2017. p. 10. Citado 4 vezes nas páginas 10, 24, 30 e 35.

MOE, N. et al. Coaching a global agile virtual team. In: *Proceedings of the 2015 IEEE 10th International Conference on Global Software Engineering*. Ciudad Real, Spain: IEEE, 2015. Citado 11 vezes nas páginas 24, 25, 26, 27, 29, 31, 34, 38, 39, 42 e 45.

MOLLÉRI, J. S.; PETERSEN, K.; MENDES, E. Survey guidelines in software engineering: An annotated review. In: *Proceedings of the 10th ACM/IEEE international symposium on empirical software engineering and measurement*. [S.l.: s.n.], 2016. p. 1–6. Citado na página 19.

NUEVO, E. del; PIATTINI, M.; PINO, F. J. Scrum-based methodology for distributed software development. In: *2011 IEEE Sixth International Conference on Global Software Engineering*. Helsinki, Finland: IEEE, 2011. p. 66–74. Citado 12 vezes nas páginas 25, 26, 27, 29, 32, 33, 34, 35, 36, 39, 43 e 45.

PAASIVAARA, M. Adopting safe to scale agile in a globally distributed organization. In: *2017 IEEE 12th International Conference on Global Software Engineering (ICGSE)*. Buenos Aires, Argentina: IEEE, 2017. p. 36–40. Citado 2 vezes nas páginas 27 e 37.

PAASIVAARA, M.; DURASIEWICZ, S.; LASSENIUS, C. Using scrum in a globally distributed project: A case study. *Softw. Process*, John Wiley & Sons, Inc., USA, v. 13, n. 6, p. 527–544, nov. 2008. ISSN 1077-4866. Citado 13 vezes nas páginas 24, 25, 26, 27, 28, 29, 33, 34, 37, 38, 39, 43 e 44.

PAASIVAARA, M.; LASSENIUS, C. Using scrum practices in gsd projects. In: ____. *Agility Across Time and Space: Implementing Agile Methods in Global Software Projects*. Berlin, Heidelberg: Springer, 2010. p. 259–278. ISBN 978-3-642-12442-6. Citado 6 vezes nas páginas 24, 25, 26, 27, 28 e 29.

PETERSEN, K. et al. Systematic mapping studies in software engineering. In: *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering*. Swindon, GBR: BCS Learning & Development Ltd., 2008. (EASE'08), p. 68–77. Citado na página 18.

PONTES, T. B.; ARTHAUD, D. D. B. Metodologias ageis para o desenvolvimento de softwares. *Ciencia E Sustentabilidade*, v. 4, n. 2, p. 173–213, 2018. Citado na página 12.

RAJPAL, M. Effective distributed pair programming. In: *Proceedings of the 13th International Conference on Global Software Engineering*. New York, NY, USA: ACM, 2018. (ICGSE '18), p. 6–10. ISBN 9781450357173. Citado 2 vezes nas páginas 27 e 37.

RALPH, P.; SHPORTUN, P. Scrum abandonment in distributed teams: A revelatory case. In: *Proceedings - Pacific Asia Conference on Information Systems, PACIS 2013*. Jeju Island, Korea: AIS, 2013. v. 101, p. 1–16. ISBN 9788995217016. Citado 6 vezes nas páginas 24, 27, 28, 29, 33 e 40.

RAMESH, B. et al. Can distributed software development be agile? *Commun. ACM*, v. 49, n. 10, p. 41–46, out. 2006. ISSN 0001-0782. Citado 6 vezes nas páginas 13, 25, 29, 30, 31 e 37.

RAMESH, B.; MOHAN, K.; CAO, L. Ambidexterity in agile distributed development: An empirical investigation. *Info. Sys. Research*, INFORMS, Linthicum, MD, USA, v. 23, n. 2, p. 323–339, jun. 2012. ISSN 1526-5536. Citado 2 vezes nas páginas 28 e 39.

RAZAVI, A. M.; AHMAD, R. Agile development in large and distributed environments: A systematic literature review on organizational, managerial and cultural aspects. In: *2014 8th. Malaysian Software Engineering Conference (MySEC)*. Langkawi, Malaysia: IEEE, 2014. p. 216–221. Citado na página 25.

RAZZAK, M. Transition from plan-driven to agile: An action research. In: *Product-Focused Software Process Improvement*. Cham: Springer, 2016. p. 746–750. Citado na página 31.

RAZZAK, M. et al. Transition from plan driven to safe®: Periodic team self-assessment. In: *Product-Focused Software Process Improvement*. Cham: Springer, 2017. p. 573–585. ISBN 978-3-319-69926-4. Citado 6 vezes nas páginas 13, 24, 26, 30, 31 e 39.

RAZZAK, M. A. et al. Scaling agile across the global organization: An early stage industrial safe self-assessment. In: *Proceedings of the 13th International Conference on Global Software Engineering*. New York, NY, USA: ACM, 2018. (ICGSE '18), p. 121–130. ISBN 9781450357173. Citado 2 vezes nas páginas 24 e 25.

RICHTER, I.; RAITH, F.; WEBER, M. Problems in agile global software engineering projects especially within traditionally organised corporations: [an exploratory semi-structured interview study]. In: *Proceedings of the Ninth International C* Conference on Computer Science & Software Engineering*. New York, NY, USA: ACM, 2016. (C3S2E '16), p. 33–43. ISBN 9781450340755. Citado 4 vezes nas páginas 25, 29, 33 e 34.

RIZVI, B.; BAGHERI, E.; GASEVIC, D. A systematic review of distributed agile software engineering. *Journal of Software: Evolution and Process*, v. 27, 06 2015. Citado 16 vezes nas páginas 12, 24, 25, 26, 27, 28, 30, 31, 32, 34, 38, 40, 41, 42, 43 e 52.

ROBINSON, P. T. Communication network in an agile distributed software development team. In: *2019 ACM/IEEE 14th International Conference on Global Software Engineering (ICGSE)*. Montreal, QC, Canada, Canada: IEEE, 2019. p. 100–104. Citado 3 vezes nas páginas 24, 25 e 27.

ROOPA, M.; KUMAR, R.; MANI, V. Transitioning from plan-driven to lean in a global software engineering organization: A practice-centric view. In: *Proceedings of the 13th International Conference on Global Software Engineering*. New York, NY, USA: ACM, 2018. (ICGSE '18), p. 1–5. ISBN 9781450357173. Citado 7 vezes nas páginas 24, 26, 27, 28, 29, 30 e 34.

SCHENK, J.; PRECHELT, L.; SALINGER, S. Distributed-pair programming can work well and is not just distributed pair-programming. In: *Companion Proceedings of the 36th International Conference on Software Engineering*. New York, NY, USA: ACM, 2014. (ICSE Companion 2014), p. 74–83. ISBN 9781450327688. Citado na página 30.

SRIRAM, R.; MATHEW, S. Global software development using agile methodologies: A review of literature. In: *2012 IEEE International Conference on Management of Innovation Technology (ICMIT)*. Sanur Bali, Indonesia: IEEE, 2012. p. 389–393. Citado 3 vezes nas páginas 24, 28 e 32.

SUNDARARAJAN, S.; BHASI, M.; VIJAYARAGHAVAN, P. K. Case study on risk management practice in large offshore-outsourced agile software projects. *IET Software*, v. 8, n. 6, p. 245–257, 2014. Citado 13 vezes nas páginas 24, 26, 27, 28, 30, 31, 32, 33, 34, 37, 38, 40 e 43.

SZABÓ, D. M.; STEGHÖFER, J.-P. Coping strategies for temporal, geographical and sociocultural distances in agile gsd: A case study. In: *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*. Montreal, QC, Canada, Canada: IEEE, 2019. p. 161–170. Citado 13 vezes nas páginas 24, 25, 26, 27, 28, 29, 30, 31, 32, 41, 42, 44 e 52.

TANNER, M.; CHIGONA, W. Towards an understanding of the contextual influences on distributed agile software development: A theory of practice perspective. *ECIS 2012 - Proceedings of the 20th European Conference on Information Systems*, v. 178, 01 2012. Citado 6 vezes nas páginas 26, 30, 31, 33, 37 e 43.

TRIPATHI, N. et al. Scaling kanban for software development in a multisite organization: Challenges and potential solutions. In: *Agile Processes in Software Engineering and Extreme Programming*. Cham: Springer, 2015. p. 178–190. Citado 10 vezes nas páginas 26, 27, 28, 30, 31, 34, 35, 37, 39 e 42.

VALLON, R. et al. Identifying critical areas for improvement in agile multi-site co-development. *ENASE 2013 - Proceedings of the 8th International Conference on Evaluation of Novel Approaches to Software Engineering*, v. 1, p. 165–172, 01 2013. Citado 10 vezes nas páginas 24, 26, 27, 29, 31, 32, 33, 34, 35 e 40.

VALLON, R. et al. Systematic literature review on agile practices in global software development. *Information and Software Technology*, v. 96, 12 2017. Citado 17 vezes nas páginas 10, 13, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 37, 39, 40, 41 e 44.

VersionOne, Inc. *14th Annual State of Agile Development Survey*. 2020. <https://stateofagile.com/#ufh-c-7027494-state-of-agile>. [Online; accessed 28-May-2020]. Citado na página 10.

WIERINGA, R. et al. Requirements engineering paper classification and evaluation criteria: A proposal and a discussion. *Requir. Eng.*, v. 11, p. 102–107, 03 2006. Citado na página 18.

A Apêndice 1

A.1 Protocolo de Revisão Sistemática

Systematic literature review protocol

1. Research questions

The main question of this study is:
RQ1: How are agile practices adopted in agile global software development teams??

2. Inclusion and exclusion criterias

2.1 Exclusion criterias

The studies that fit in one of these criteria will be excluded:

- Studies not written in English;
- Documents that are editorials, tutorials, key-note speech, white papers, thesis, dissertations, technical reports, books;
- Short papers (<= 4 pages).

2.2 Inclusion criterias

The studies that fit in these criteria will be included:

- Studies that were published in journals and peer-reviewed conferences;
- Studies directly related to the research questions;
- Studies that are available, via the university library services, to the authors during the time of search (Universidade Federal Rural de Pernambuco) or available on the web.

When multiple publications of the same study in different journals or conferences occurs, both versions of the study will be reviewed, but only the first study published will be included in our SRL

3. Search Strategy

Our search was direct for all text to maximize the discovery of relevant papers

3.1 Search String

Our search string looks for synonyms, variants, and abbreviations of the main themes. The search string looks for agile AND global software development (VALLON et al., 2017).

We used the searches to identify a set of relevant papers that should be matched by the actual search. We started by examining top-ranked hits by trivial keywords that the more complex final search string might miss. Initial searches were made using keywords that were as general as possible, including ``agile methods'' and ``global software development''. Based on these preliminary searches, we selected the terms for the string. Agile practices

were limited to Scrum, XP, Safe, Lean Software Development, and pair programming, intending to cover the most common ones, which are mostly used in practice.

Terms used to build the search string:

Identified terms	Terms to be used	
Global Software Development	● global software engineering ● global software development ● distributed software engineering ● distributed software development ● GSE ● GSD ● distributed team	● global team ● dispersed team ● spread team ● virtual team ● offshore ● outsource
Agile	● Agile ● Scrum ● extreme programming ● pair programming ● hybrid	● lean development ● lean software development ● SAFe ● Scaled Agile Framework

Our main string combines the agile terms with global software development terms, resulting in:

Main search String
(“Agile” OR “scrum” OR “extreme programming” OR “pair programming” OR “hybrid” OR “lean development” OR “lean software development” OR “SAFe” OR “Scaled Agile Framework”) AND (“global software engineering” OR “global software development” OR “distributed software engineering” OR “distributed software development” OR “GSE” OR “GSD” OR “distributed team” OR “global team” OR “dispersed team” OR “spread team” OR “virtual team” OR “offshore” OR “outsource”)

4. Literature Search

The chosen search engines for this study were select following two criteria:

- Availability of the studies on the internet;
- Importance and relevance of the research source.

Following these criteria, five search engines were chosen:

Search Engine	Site
IEEE Xplore Digital Library	http://ieeexplore.ieee.org/
ACM Digital Library	http://portal.acm.org/
Scopus	http://www.scopus.com/

SpringerLink	http://www.springerlink.com/
Wiley InterScience	https://onlinelibrary.wiley.com

5. Quality Evaluation

Nº	Pontuação / Questão	1pt	0,5pt	0pt
1	Is there a clear definition of the study objectives?	The objectives of the study are clearly defined.	The objectives of the study are not clearly defined, but it is implicitly in the text.	The objectives of the study are not defined.
2	Is there a clear definition of the justifications of the study?	The justification for the study is clearly defined.	The justification for the study is loosely defined.	The justification for the study is not mentioned.
3	Is there a theoretical background about the topics of the study?	The theoretical background is consistent.	The theoretical background is weak.	The study does not have a theoretical background.
4	Is there a clear definition of the research question (RQ) and/or the hypothesis of the study?	The RQ / study hypotheses are clearly defined.	The RQ / study hypotheses are not clearly defined, but are implicit in the text.	The RQ / study hypotheses are not defined.
5	Is there an adequate description of the context in which the research was carried out?	The context in which the study is carried out is described.	The context in which the study is carried out is partially defined.	The context in which the study is carried out is not described.
6	Are data collection methods used and described appropriately?	The data collection methods used are appropriated and were described.	The data collection methods used are not appropriated or were partially described.	The data collection methods used are not appropriated and were not described.

7	Is there an adequate description of the sample used and the methods for identifying and recruiting the sample?	The sample and methods to identify the sample were described.	The sample and methods to identify the sample were partially described.	The sample and methods to identify the sample were not described.
8	Are there an adequate description of the methods used to analyze data and appropriate methods for ensuring that data analysis was grounded in the data?	The study clearly describes the method to analyze data and the results are according to the data analysis method chosen.	The study partially describes the method to analyze data or the results are partially according to the data analysis method chosen.	The study does not describe the method to analyze data and the results are not according to the data analysis method chosen.
9	Are clear answers or justifications about RQ/hypothesis provided by the study?	RQ / study hypotheses were answered successfully, or justified.	Some of the RQ / study hypotheses were answered / justified.	The RQ / hypotheses were not answered and there is no justification for this fact.
10	Are clearly stated findings with credible results provided by the study?	The study published original results with findings in a clear way.	The study publishes original results without clearness.	The study does not publish original results.
11	Are justified conclusions provided by the study?	The study provides justified conclusions.	The study provides justified conclusions implicitly.	The study does not provide justified conclusion
12	Is discussion about validity threats provided by the study?	There is a specific section to discuss the validation threats of the study.	The validation threats are discussed through the text or partially.	The author does not discuss the validation threats of the study.conclusions.

A.2 Tabela dos estudos de pesquisa

Referências	Título	Local de publicação	Base
RAJPAL, 2018	Effective Distributed Pair Programming	Proceedings of the 13th International Conference on Global Software Engineering	ACM
ROBINSON, 2019	Communication Network in an Agile Distributed Software Development Team	2019 ACM/IEEE 14th International Conference on Global Software Engineering (ICGSE)	IEEE
RAZZAK et al., 2018	Scaling Agile across the Global Organization: An Early Stage Industrial SAFe Self-Assessment	{Proceedings of the 13th International Conference on Global Software Engineering	ACM
LOUS et al., 2018b	Virtual by Design: How a Work Environment Can Support Agile Distributed Software Development	Proceedings of the 13th International Conference on Global Software Engineering	ACM
ALTAF et al., 2019	A Systematic Literature Review on Factors Impacting Agile Adaptation in Global Software Development	Proceedings of the 2019 7th International Conference on Computer and Communications Management	ACM
SZABÓ; STEGHÖFER, 2019	Coping Strategies for Temporal, Geographical and Sociocultural Distances in Agile GSD: A Case Study	2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)	IEEE
LOUS; KUHRMANN; TELL, 2017	Is Scrum Fit for Global Software Engineering?	Proceedings of the 12th International Conference on Global Software Engineering	IEEE
PAASIVAARA, 2017	Adopting SAFe to Scale Agile in a Globally Distributed Organization	2017 IEEE 12th International Conference on Global Software Engineering (ICGSE)	IEEE
TAYLOR et al., 2006	Do Agile GSD Experience Reports Help the Practitioner?	Proceedings of the 2006 International Workshop on Global Software Development for the Practitioner	ACM
HOSSAIN; BANNERMAN; JEFFERY, 2011b	Towards an Understanding of Tailoring Scrum in Global Software Development: A Multi-Case Study	Proceedings of the 2011 International Conference on Software and Systems Process	ACM
LOUS et al., 2018a	From Scrum to Agile: A Journey to Tackle the Challenges of Distributed	Proceedings of the 2018 International Conference on Software and System Process	ACM

	Development in an Agile Team		
ROOPA; KUMAR; MANI, 2018	Transitioning from Plan-Driven to Lean in a Global Software Engineering Organization: A Practice-Centric View	Proceedings of the 13th International Conference on Global Software Engineering	ACM
SCHENK; PRECHELT;SA LINGER, 2014	Distributed-Pair Programming Can Work Well and is Not Just Distributed Pair-Programming	Companion Proceedings of the 36th International Conference on Software Engineering	ACM
GUPTA; JAIN; SINGH, 2018	Challenges in Scaling up a Globally Distributed Legacy Product: A Case Study of a Matrix Organization	Proceedings of the 13th International Conference on Global Software Engineering	ACM
LAL; CLEAR, 2018	Enhancing Product and Service Capability through Scaling Agility in a Global Software Vendor Environment	Proceedings of the 13th International Conference on Global Software Engineering	ACM
RAMESH etal., 2006	Can Distributed Software Development Be Agile?	Communications of the ACM	ACM
RICHTER; RAITH; WEBER, 2016	Problems in Agile Global Software Engineering Projects Especially within Traditionally Organised Corporations: [An Exploratory Semi-Structured Interview Study]	Proceedings of the Ninth International C* Conference on Computer Science \& Software Engineering	ACM
BATRA, 2009	Modified Agile Practices for Outsourced Software Projects	Communications of the ACM	ACM
LEE; JUDGE; MCCRICKARD, 2011	Evaluating EXtreme Scenario-Based Design in a Distributed Agile Team	CHI '11 Extended Abstracts on Human Factors in Computing Systems	ACM
SIEVIKORTE; RICHARDSON; BEECHAM, 2019	Software Architecture Design in Global Software Development: An Empirical Study	Journal of Systems and Software	IEEE
ALSAQAF; DANEVA;WIERI NGA, 2017	Quality Requirements in Large-Scale Distributed Agile Projects -- A Systematic Literature Review	International Working Conference on Requirements Engineering: Foundation for Software Quality	Springer

BJØRN; SØDERBERG; KRISHNA, 2019	Translocality in Global Software Development: the Dark Side of Global Agile	Human-Computer Interaction	IEEE
HOSSAIN, 2019	Challenges and Mitigation Strategies in Reusing Requirements in Large-Scale Distributed Agile Software Development: A Survey Result	Intelligent Computing-Proceedings of the Computing Conference	Springer
VALLON et al., 2017	Systematic Literature Review on Agile Practices in Global Software Development	Information and Software Technology	IEEE
GERVIGNY; NAGOWAH, 2017	Knowledge sharing for agile distributed teams: A case study of Mauritius	2017 International Conference on Infocom Technologies and Unmanned Systems (Trends and Future Directions) (ICTUS)	IEEE
Khmelevsky; Li; Madnick, 2017	Khmelevsky, Youry and Li, Xitong and Madnick, Stuart	2017 Annual IEEE International Systems Conference (SysCon)	IEEE
KAUSAR; ALYASIRI, 2017	Using Distributed Agile Patterns for Supporting the Requirements Engineering Process	Requirements Engineering for Service and Cloud Computing	Springer
LAUREN, 2015	Mapping the Workspace of a Globally Distributed "Agile" Team	International Journal of Sociotechnology and Knowledge Development	ACM
USMAN; BRITTO, 2016	Effort Estimation in Co-located and Globally Distributed Agile Software Development: A Comparative Study	2016 Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement (IWSM-MENSURA)	IEEE
BANIJAMALIet al., 2017	Empirical Investigation of Scrumban in Global Software Development	Model-Driven Engineering and Software Development	Springer
RAZZAK et al., 2017	Transition from Plan Driven to SAFe®: Periodic Team Self-Assessment	Product-Focused Software Process Improvement	Springer
GONÇALVES et al., 2017	Using Agile Methods in Distributed Software Development Environments	Communications in Computer and Information Science	Springer

MODI; ABBOTT; COUNSELL, 2017	Exploring the Emergence of Collaborative Practices in Globally Distributed Agile Software Development	AMCIS	Springer
RAZZAK, 2016	Transition from Plan-Driven to Agile: An Action Research	Product-Focused Software Process Improvement	Springer
KHAN; AKBAR; WONG, 2016	A Study on Global Software Development (GSD) and Software Development Processes in Malaysian Software Companies	Journal of Telecommunication, Electronic and Computer Engineering	Scopus
GUPTA; MANIKREDDY, 2015	Challenges in Adapting Scrum in Legacy Global Configurator Project	2015 IEEE 10th International Conference on Global Software Engineering	IEEE
MOE et al., 2015	Coaching a Global Agile Virtual Team	Proceedings of the 2015 IEEE 10th International Conference on Global Software Engineering	IEEE
BRITTO; MENDES; BÖRSTLER, 2015	An Empirical Investigation on Effort Estimation in Agile Global Software Development	2015 IEEE 10th International Conference on Global Software Engineering	IEEE
SHARMA; KAUR;KAUR, 2015	Communication understandability enhancement in GSD	2015 International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE)	IEEE
TRIPATHI et al., 2015	Scaling Kanban for Software Development in a Multisite Organization: Challenges and Potential Solutions	Agile Processes in Software Engineering and Extreme Programming	Springer
RIZVI; BAGHERI; GASEVIC, 2015	A systematic review of distributed Agile software engineering	Journal of Software: Evolution and Process	Scopus
ESTáCIO;PRIKLADNICKI, 2014	A Set of Practices for Distributed Pair Programming	ICEIS	Scopus
BRITTO; USMAN; MENDES, 2014	Effort Estimation in Agile Global Software Development Context	Lecture Notes in Business Information Processing	Springer
HUBER; DIBBERN, 2014	How Collaboration Software Enables Globally Distributed Software Development Teams to Become Agile - An Effective Use Perspective	Governing Sourcing Relationships. A Collection of Studies at the Country, Sector and Firm Level	Springer

SUNDARARAJAN; BHASI; VIJAYARAGHAVAN, 2014	Case study on risk management practice in large offshore-outsourced Agile software projects	IET Software	Wiley
RAZAVI; AHMAD, 2014	Agile development in large and distributed environments: A systematic literature review on organizational, managerial and cultural aspects	2014 8th. Malaysian Software Engineering Conference (MySEC)	IEEE
HAMID, 2013	Upgrading distributed agile development	2013 International Conference on Computing, Electrical and Electronic Engineering (ICCEEE)	IEEE
VALLON et al., 2013	Identifying critical areas for improvement in agile multi-site co-development	ENASE 2013 - Proceedings of the 8th International Conference on Evaluation of Novel Approaches to Software Engineering	Scopus
RALPH; SHPORTUN, 2013	Scrum Abandonment in Distributed Teams: A Revelatory Case	Proceedings - Pacific Asia Conference on Information Systems, PACIS 2013	Scopus
LI; MäDCHE, 2013	Formulating Effective Coordination Strategies in Agile Global Software Development Teams	Digital Innovation in the Service Economy. 33th International Conference on Information Systems (ICIS)	Scopus
SRIRAM; MATHEW, 2012	Global software development using agile methodologies: A review of literature	2012 IEEE International Conference on Management of Innovation Technology (ICMIT)	IEEE
DORAIRAJ; NOBLE; MALIK, 2012	Understanding Team Dynamics in Distributed Agile Software Development	Agile Processes in Software Engineering and Extreme Programming	Springer
TANNER; CHIGONA, 2012	Towards an understanding of the contextual influences on distributed agile software development: A theory of practice perspective	ECIS 2012 - Proceedings of the 20th European Conference on Information Systems	Scopus
RAMESH; MOHAN; CAO, 2012	Ambidexterity in Agile Distributed Development: An Empirical Investigation	Information Systems Research	Scopus
DUMITRIU et al., 2011	Issues and strategy for agile global software development adoption	Proceedings of the World Multiconference on Applied Economics, Business and Development	Scopus

HOSSAIN; BANNERMAN; JEFFERY, 2011a	Scrum Practices in Global Software Development: A Research Framework	Product-Focused Software Process Improvement	Springer
DORAIRAJ; NOBLE; MALIK, 2011	Bridging Cultural Differences: A Grounded Theory Perspective	Proceedings of the 4th India Software Engineering Conference	ACM
HILLEGERSBERG;LIGTENBERG; AYDIN, 2011	Getting Agile Methods to Work for Cordys Global Software Product Development	New Studies in Global IT and Business Service Outsourcing	Springer
PAASIVAARA; LASSENIUS, 2010	Using Scrum Practices in GSD Projects	Agility Across Time and Space: Implementing Agile Methods in Global Software Projects	Springer
MARUPING, 2010	Implementing Extreme Programming in Distributed Software Project Teams: Strategies and Challenges	Agility Across Time and Space: Implementing Agile Methods in Global Software Projects	Springer
AVRITZER; BRONSARD; MATOS, 2010	Improving Global Development Using Agile	Agility Across Time and Space: Implementing Agile Methods in Global Software Projects	Springer
ANSARI; SHARAFI; NEMATBAKSSH, 2010	A method for requirements management in distributed extreme programming environment	Journal of Theoretical and Applied Information	ACM
HOSSAIN; BABAR; VERNER, 2009a	How Can Agile Practices Minimize Global Software Development Co-ordination Risks?	Software Process Improvement	Springer
HOSSAIN; BABAR; VERNER, 2009b	Towards a Framework for Using Agile Approaches in Global Software Development	Product-Focused Software Process Improvement	Springer
PAASIVAARA; DURASIEWICZ; LASSENIUS, 2008	Using Scrum in a Globally Distributed Project: A Case Study	Software Process: Improvement and Practice	Wiley
WAHYUDIN et al., 2008	In-Time Role-Specific Notification as Formal Means to Balance Agile Practices in Global Software Development Settings	Balancing Agility and Formalism in Software Engineering	Springer

HOLE;MOE, 2008	A Case Study of Coordination in Distributed Agile Software Development	Software Process Improvement	Springer
CHO, 2007	Distributed Scrum for Large-Scale and Mission-Critical Projects	AMCIS	Scopus
KUSSMAUL; JACK; SPONSLER, 2004	Outsourcing and Offshoring with Agility: A Case Study	Extreme Programming and Agile Methods - XP/Agile Universe 2004	Springer
BASS, 2015	How product owner teams scale agile methods to large distributed enterprises	Empirical Software Engineering	Springer
LEE;YONG, 2009	Distributed agile: project management in a global environment	Empirical Software Engineering	Scopus
DINGSøYR et al., 2017	Exploring software development at the very large-scale: a revelatory case study and research agenda for agile method adaptation	Empirical Software Engineering	Springer
JALALI; WOHLIN, 2010	Agile Practices in Global Software Engineering - A Systematic Map	2010 5th IEEE International Conference on Global Software Engineering	IEEE
JHA; VILARDELL;NARAYAN, 2016	Scaling Agile Scrum Software Development: Providing Agility and Quality to Platform Development by Reducing Time to Market	2016 IEEE 11th International Conference on Global Software Engineering (ICGSE)	IEEE
HOSSAIN; BABAR; PAIK, 2009	Using Scrum in Global Software Development: A Systematic Literature Review	2009 Fourth IEEE International Conference on Global Software Engineering	IEEE
NUEVO; PIATTINI; PINO, 2011	Scrum-based Methodology for Distributed Software Development	2011 IEEE Sixth International Conference on Global Software Engineering	IEEE
ILYAS; KHAN; RASHID, 2020	Empirical Validation of Software Integration Practices in Global Software Development	SN Computer Science	Springer
QAHTANI, 2020	An Empirical Study of Agile Testing in A Distributed Software Development Project	Proceedings of the 2020 3rd International Conference on Geoinformatics and Data Analysis	ACM

SINHA; SHAMEEM; KUMAR, 2020	SWOT, Agile development, Scaling, Global software development	Proceedings of the 13th Innovations in Software Engineering Conference on Formerly Known as India Software Engineering Conference	ACM
STRAY; MOE, 2020	Understanding coordination in global software engineering: A mixed-methods study on the use of meetings and Slack	Journal of Systems and Software	ACM
ESTEKI TAGHI JAVDANI GANDOMANI, 2020	A risk management framework for distributed scrum using prince2 methodology	Bulletin of Electrical Engineering and Informatics	ACM
KAMAL;ZHANG; AKBAR, 2020	Toward successful agile requirements change management process in global software development: A client-vendor analysis	IET Software	Scopus
SHAMEEM et al., 2020	Taxonomical classification of barriers for scaling agile methods in global software development environment using fuzzy analytic hierarchy process	Applied Soft Computing Journal	Scopus
HIDAYATI; BUDIARDJO; PURWANDARI, 2020	Hard and soft skills for scrum global software development teams	ACM International Conference Proceeding Series	Scopus
HOSSAIN et al., 2020	Requirements Re-usability in Global Software Development: A Systematic Mapping Study	Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)	Scopus
SMITE; GONZALEZHUERTA; MOE, 2020	Evaluating EXtreme Scenario-Based Design in a Distributed Agile Team	Lecture Notes in Business Information Processing	Scopus

B Apêndice 2 – Protocolo do Survey

B.1 Perguntas

B.1.1 Qual dos seguintes frameworks você geralmente usa?

- Processo Clássico em Cascata.
- Família Crystal.
- DevOps.
- Design orientado por domínio.
- Método de Desenvolvimento de Sistemas Dinâmicos (DSDM).
- Programação Extrema (XP).
- Desenvolvimento baseado em recursos (FDD).
- Desenvolvimento Iterativo.
- Kanban.
- Scrum em larga escala. (LESS)
- Lean Software Development.
- Model Driven Architecture
- Nexus.
- Processo de Software Personalizado.
- Stage-gate model
- PRINCE2.
- Processo Unificado da Racional (RUP).
- Scaled Agile Framework (SAFe)
- Scrum.
- ScrumBan.
- Modelo Espiral.

- Análise de Sistemas Estruturados e Método de Projeto (SSADM).
- Vshaped Process (VModel).

B.1.2 Quais abordagens de desenvolvimento de software você utiliza no seus projetos?

- Abordagens totalmente ágeis.
- Principalmente abordagens ágeis.
- Balanceado entre abordagens ágeis e abordagens tradicionais.
- Principalmente abordagens tradicionais.
- Abordagens totalmente tradicionais.

B.1.3 Com que frequência você utiliza as seguintes práticas?

- Reunião diária
- Práticas de comunicação
- Planejamento
- Auto Gerenciamento
- Scrum de scrums
- Histórias de usuários
- Revisão de sprint
- Reunião retrospectiva
- Backlog do produto
- Visitas entre locais de trabalho
- Sprint
- Gestão de pendências
- Programação em par
- Quadro kanban
- Integração contínua

- Burndown
- Projete a equipe
- Gerenciamento de tarefas
- Coaching
- Documentação necessária
- Sincronizar horas de trabalho
- Equipe reunida no início do projeto
- Reunião de estimativa
- TDD
- Implantação contínua
- Wiki do projeto
- Demonstração do sistema
- Automação de teste
- Definição de pápeis da equipe
- Revisão de código
- Reuniões informais
- Gerenciar expectativas do cliente
- Jogo de planejamento
- Feedbacks frequentes
- Arquitetura ágil
- Entrega contínua
- Padrões de codificação
- Propriedade do código coletivo
- Missão e visão compartilhada
- Refactoring
- Alternar equipe entre locais

- Bug tracker
- Design simples
- Sprint de tamanho fixo
- Gestão de testes
- Documentar lições aprendidas
- Roadmap planning
- Testes de aceitação
- Escalonamento do Kanban
- Expandir respons. das equipes
- Colaboração entre equipes

B.1.4 Quantos anos de experiência você tem em desenvolvimento de software e sistemas?

- Menos de 1 ano.
- 1 - 2 anos.
- 3 - 5 anos.
- 6 - 10 anos.
- Mais de 10 anos.

B.1.5 Qual é o tamanho da sua empresa em quantidade de funcionários?

- Micro (< 10 funcionários)
- Pequena (11 - 50 funcionários)
- Médio (51 - 250 funcionários)
- Grande (251 - 2499 funcionários)
- Muito grande (> 2500 funcionários)

B.1.6 Qual é a principal área de negócios da sua empresa?

- Área de negócios: opção residual. (negativa) ou número de opções selecionadas
- Desenvolvimento de software. (software personalizado, ou seja, soluções individuais)
- Desenvolvimento de software. (software padrão, por exemplo, SAP, Office)
- Desenvolvimento de sistema. (hardware e software, por exemplo, sistemas incorporados)
- Consultoria / Suporte à Gestão de Projetos.
- Consultoria, treinamento e serviços de TI.
- Pesquisa e Desenvolvimento.

B.1.7 Sua empresa atua de maneira (globalmente) distribuída?

- Não.
- Sim, nacionalmente. (mesmo país)
- Sim, regionalmente. (mesmo continente)
- Sim, globalmente.

B.1.8 Qual é a principal função que você desempenha nesta empresa?

- Arquiteto.
- CIO / CTO.
- Desenvolvedor.
- Gerente de Produto.
- Gerente de Projeto.
- Gerente de Qualidade.
- Scrum Master.
- Testador.
- Treinador.