DOI: 10.5748/19CONTECSI/PSE/PRM/6936

A PROPOSAL FOR REUSING LESSONS LEARNED IN SCRUM PROJECTS

PROPOSTA PARA REUSO DE LIÇÕES APRENDIDAS EM PROJETOS SCRUM

Elton Grottoli Lima ; https://orcid.org/0000-0002-2168-5374
Instituto de Pesquisas Tecnológicas de São Paulo - IPT

Vagner Luiz Gava ; https://orcid.org/0000-0001-5965-957X
Instituto de Pesquisas Tecnológicas de São Paulo - IPT

A PROPOSAL FOR REUSING LESSONS LEARNED IN SCRUM PROJECTS

ABSTRACT

Research reveals that a considerable number of IT projects fail to meet their schedule and/or cost estimates. One of the reasons for this low success rate is that organizations are unable to reuse lessons learned in projects or are doing so in an incomplete or inefficient way. The goal of this study is to build a process for estimulating the reuse of lessons learned in Scrum projects. A systematic literature review is performed, which reveals the most popular techiniques used for conducting Retrospectives have flaws that inhibit the learning process. By reintroducing techniques that will mitigate these flaws, a process is presented with the expected outcome of enabling the reuse of lessons learned during the project and, as a result, to help the team to avoid past mistakes and distribute knowledge in a more homogeneous way among team members.

Keywords: Scrum, knowledge management, reuse, lessons learned, retrospective, tool, framework

PROPOSTA PARA REUSO DE LIÇÕES APRENDIDAS EM PROJETOS SCRUM

RESUMO

Pesquisas revelam que mais da metade dos projetos de TI falham em sua implementação, superando seu planejamento inicial de prazos e custos. Uma das causas subjacentes à esta baixa taxa de sucesso é a incapacidade das organizações em reutilizar as lições aprendidas em projetos, ou fazê-lo de forma incompleta ou ineficaz. O objetivo deste artigo é construir um processo que estimule o reuso de lições aprendidas em projetos Scrum. É conduzida uma revisão sistemática da literatura, cujo resultado revela que a cerimônia de Retrospectiva, em suas formas mais comuns, apresenta falhas de condução que inibem o processo de aprendizado no projeto. Com a reintrodução de práticas que endereçem estas falhas, apresenta-se um processo cujo resultado esperado é facilitar o reuso das lições aprendidas no projeto e, com isso, evitar a repetição de erros passados e distribuir o conhecimento de forma mais homogênea na equipe.

Palavras-chave: Scrum, gestão de conhecimento, reuso, lições aprendidas, retrospectiva, ferramenta

1. INTRODUÇÃO

1.1 APRESENTAÇÃO E CONTEXTUALIZAÇÃO

Dados disponíveis na literatura sobre as taxas de sucesso na implementação de projetos de TI revelam números preocupantes para as organizações que lidam com estes projetos. O Chaos Report, que analisa informações sobre uma base de 50.000 projetos, revela que 60% destes foram entregues com atraso e 56% estouraram o orçamento previsto (STANDISH GROUP, 2015). Em outro estudo, conduzido pela empresa Cisco, é revelado que a cada quatro projetos relacionados a Internet of Things (IoT), em média três deles falham em sua implementação (CISCO, 2017a), com uma taxa de sucesso reportada de meros 26% (CISCO, 2017b). Este estado de coisas sugere a necessidade de um aprofundamento no estudo das causas que provocam o insucesso de projetos de TI, bem como indica uma área de oportunidade para a pesquisa de melhorias em processos de desenvolvimento que possam mitigar estas causas.

Estudos como o de Williams (2008), Adnan e Afzal (2017), e Darfeuille (2017) apontam que uma das causas para o insucesso de um projeto é, de forma genérica, a inadequação de seu processo de gestão do conhecimento, cuja aplicação deveria facilitar a reutilização de lições aprendidas. Pode-se dizer que as organizações sabem mais do que pensam, mas falham em criar uma memória organizacional (DARFEUILLE, 2017), o que viabilizaria o reuso sistemático de conhecimento. Esta afirmação também encontra respaldo no trabalho de Simkonis e Skyrius (2013), onde se adverte que sem o uso de lições aprendidas, uma organização irá repetir erros passados, podendo retroceder rapidamente à fase de imaturidade em projetos.

O reuso de lições aprendidas também aparece como fator crítico de sucesso nas estatísticas já mencionadas anteriormente. Por exemplo, no estudo conduzido pela Cisco (2017), 64% dos respondentes concordaram sobre a afirmação que “aprender com os erros em projetos anteriores ajuda a acelerar a implementação de projetos” (CISCO, 2017a). Esse resultado alinha-se ao discutido no trabalho de Adnan e Afzal (2017), onde se aponta que a incapacidade em reutilizar o conhecimento de projetos anteriores é uma das causas para o insucesso dos projetos de software. Mais ainda, Williams (2008) constatou que a maior parte dos projetos não é revisada de nenhuma forma, e quando o são, os métodos utilizados não oferecem entendimento real que possa ser incorporado como aprendizado organizacional. Em resumo, estes trabalhos convergem para o entendimento que as lições aprendidas em um projeto não são capturadas ou reutilizadas de forma adequada. De forma geral, o baixo reuso de conhecimento em desenvolvimento ágil de software é um dos maiores problemas nas organizações (INDUMINI; VASANTHAPRIYAN, 2018). Esta convergência de estudos recentes para o problema do reuso justifica a sua relevância e atualidade, assim como os impactos produtivos e financeiros reportados nas pesquisas já citadas.

Este, entretanto, não é um problema novo. Ao contrário, diversos modelos de gestão de conhecimento foram propostos em meados dos anos 1990 tentando, entre outros objetivos, endereçar o problema do reaproveitamento. De fato, é possível encontrar na literatura mais de 150 modelos de gestão de conhecimento (TENORIO *et al*., 2020). Essa multiplicidade de sistemas para a gestão do conhecimento em projetos é um indicador de que este ainda é um problema em aberto ou, ao menos, de que não existe um consenso na comunidade de projetos sobre um paradigma que o solucione.

Considerando-se o papel importante que a reutilização de lições aprendidas possui como fator de sucesso em projetos, o objetivo deste artigo é construir, com base na literatura, um processo que habilite este reuso de forma integrada aos processos ágeis já tão disseminados nas organizações. Dentro da proposta ágil, há de se estabelecer oportunidades

para a revisão de lições aprendidas no contexto das cerimônias existentes. Fomentar esta atividade dentro dos processos iterativos é fundamental para que o reuso de lições aprendidas seja sistematizado por times ágeis.

Com este objetivo, formula-se a pergunta de pesquisa que norteia este artigo: *Como desenvolver um processo que habilite o reuso de lições aprendidas em projetos de desenvolvimento ágil de software?* Respondê-la será o propósito das seções seguintes. Espera-se que os resultados gerados por esta pesquisa possam auxiliar as organizações a habilitar o reuso sistemático de lições aprendidas em projetos, e assim aumentar suas taxas de sucesso.

As seções seguintes estão organizadas da seguinte maneira: Na seção 2 é apresentada a fundamentação teórica que servirá de base para o entendimento do processo proposto. Em seguida, a seção 3 apresenta o método de pesquisa utilizado, cujos resultados serão analisados na seção 4. Por fim, na seção 5 são apresentadas as conclusões deste trabalho, bem como suas limitações e sugestões para estudos futuros.

2. FUNDAMENTAÇÃO TEÓRICA

2.1 SCRUM

A publicação do Manifesto Ágil em 2001 marcou o que se mostraria como o início de um novo paradigma no desenvolvimento de software, baseado em entregas frequentes e incrementais de funcionalidades, que realimentam um ciclo de feedback constante em direção a um ideal de produto. Neste formato, as mudanças são bem-vindas pois ajudam a direcionar o desenvolvimento do produto ao seu estado ótimo, abandonando-se atividades que não adicionam valor ao processo.

Este modelo se opõe de maneira frontal ao paradigma então vigente, que colocava as etapas de desenvolvimento em uma sequência fixa, iniciando-se com o planejamento, e a partir daí seguiam-se desenvolvimento, testes e implantação. Mudanças após a etapa de planejamento exigiam esforço para serem incorporadas e, via de regra, impactavam de forma indesejada o plano original.

Os autores do Manifesto Ágil propuseram uma filosofia baseada em valores que suportaria novas formas de desenvolver software (BECK *et al.*, 2001). Estes valores são:

- Indivíduos e interações têm mais valor que processos e ferramentas
- Software funcional tem mais valor do que uma documentação abrangente
- Colaboração com o cliente tem mais valor do que negociar o contrato
- Responder à mudança tem mais valor do que seguir um plano

Da aplicação destes valores, surgiram alguns *frameworks* de desenvolvimento ágil, dos quais o mais utilizado é o Scrum, como informa o relatório State of Agile.

O Scrum Guide define o Scrum como um framework para o desenvolvimento de soluções adaptativas para problemas complexos (SCHWABER; SUTHERLAND, 2020). Ele é formado por cinco eventos e três artefatos:

Eventos do Scrum:

- Sprint: O arcabouço de tempo no qual os demais eventos do Scrum acontecem. Uma Sprint tem uma duração fixa de no máximo um mês. As Sprints seguem-se em sequência, uma após a outra, ininterruptamente.

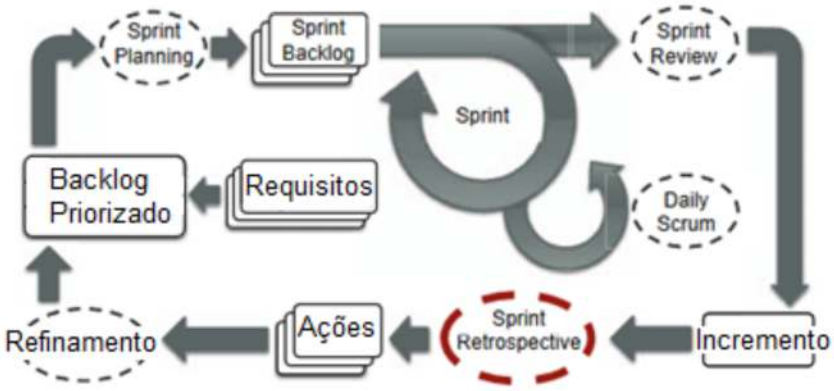
- Sprint Planning: É o evento que inicia uma Sprint, na qual o time se reúne para planejar o trabalho que será executado dentro da Sprint. Nesta reunião se define um objetivo para a Sprint.
- Daily Scrum: Um evento rápido e diário, com duração de quinze minutos, com o objetivo de inspecionar o progresso do time em direção ao objetivo da Sprint.
- Sprint Review: Um evento que se realiza ao final da Sprint, onde o time deve inspecionar o resultado da Sprint e determinar adaptações futuras baseados no feedback desta inspeção.
- Sprint Retrospective: O último evento da Sprint tem como objetivo planejar maneiras de aumentar a qualidade e a efetividade do time.

Artefatos do Scrum:

- Product Backlog: É uma lista de itens, priorizados pelo seu valor para o produto. O Product Backlog é a fonte única do trabalho que será executado pelo time Scrum. Desta lista são selecionados os itens a incluir na Sprint, durante a Sprint Planning.
- Sprint Backlog: Este artefato é composto por um objetivo de Sprint, um conjunto de itens selecionados do Product Backlog para serem executados durante a Sprint, e um plano para implementá-los.
- Incremento: É uma unidade de valor do produto, independente, funcional, e que pode ser integrada de forma aditiva a Incrementos anteriores, em um conjunto que opere de forma coesa.

A Figura 1 apresenta uma visão geral de como os eventos e artefatos do Scrum se integram em um processo iterativo de desenvolvimento:

Figura 1 - Eventos e artefatos do Scrum



Fonte: Adaptado de Matthies (2020)

2.2 CONHECIMENTO EM PROJETOS

A gestão de conhecimento em não é um tema novo. Pelo contrário, Tenório *et al.* (2020) cita a existência de mais de cento e cinquenta modelos de gestão de conhecimento, sendo que alguns ainda influentes datam dos anos 90.

Em Andryiani (2017), como resultado de uma revisão sistemática da literatura, aponta-se a existência de três tipos de conhecimento específicos em projetos de software:

- Conhecimento de projeto: Representado por cronogramas, pelo estado do time em relação ao seu avanço no projeto e pelo seu planejamento;
- Conhecimento de produto: Na forma de requisitos e arquiteturas;
- Conhecimento de processo: Representado pelas técnicas de codificação e de trabalho em equipe.

Figura 2 – Relações entre os conceitos de projeto, processo e produto

Fonte: Adaptado de (PMI, 2017)

Dingsøyr e Hanssen (2003) apresentam dois conceitos fundamentais para o entendimento da dinâmica do reuso de conhecimento em projetos (e, portanto, de lições aprendidas). O primeiro deles é a separação do conhecimento em dois tipos:

- O conhecimento **tácito** é aquele que os seres humanos possuem, mas são incapazes de representar;
- O conhecimento **explícito** é aquele possível de representar como um processo ou outra forma de documentação.

O segundo conceito são os modos pelos quais o conhecimento é convertido entre um tipo e outro. O Quadro 1 resume os modos de conversão:

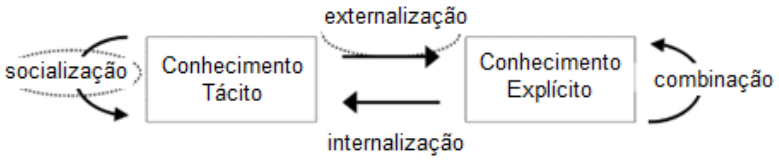
Quadro 1 – Modos de conversão do conhecimento

De	Para	Modo
Tácito	Explícito	**Externalização:** Pelo diálogo e reflexão coletiva
Explícito	Explícito	**Combinação:** Através da reorganização e junção de documentos
Explícito	Tácito	**Internalização**: Pela leitura e estudo individual
Tácito	Tácito	**Socialização:** Pelo ato de observar e atuar em conjunto

Fonte: Elaborado pelo autor

E a Figura 3 sintetiza o processo:

Figura 3 – Conversão entre conhecimento tácito e explícito



Fonte: (DINGSØYR; HANSSEN, 2003)

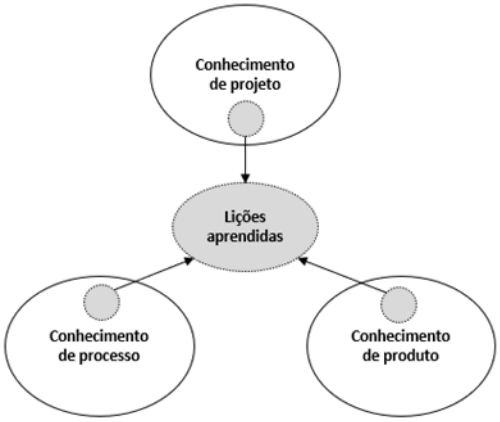
Definido então o conceito de conhecimento em projetos, é importante salientar que o uso de práticas ágeis incentiva uma relação diferente com este conhecimento. Se outrora o conhecimento do projeto estava, ao menos parcialmente, capturado em uma documentação mais ou menos extensa (conhecimento explícito), o Scrum Guide adverte sobre sua filosofia empirista no sentido que "o conhecimento vem da experiência e tomada de decisão baseada no que é observado" (SCHWABER; SUTHERLAND, 2020). No Manifesto Ágil, é possível notar tanto a despriorização do conhecimento explícito apontada no valor "software funcional sobre documentação abrangente" (BECK *et al.*, 2001), quanto o direcionamento de externalizar e socializar um conhecimento prioritariamente tácito no valor "indivíduos e interações sobre processos e ferramentas" (BECK *et al.*, 2001).

2.3 LIÇÕES APRENDIDAS EM PROJETOS

O termo "lições aprendidas" aparece com frequência na literatura de projetos sem uma definição adequada, como se encerrasse em si mesmo um significado arquetípico. No contexto deste trabalho, porém, torna-se necessário definir o termo propriamente, de maneira a delimitá-lo como objeto de investigação científica. Na sexta edição do Guia do Conhecimento em Gerenciamento de Projetos (Guia PMBOK), o termo "lições aprendidas" é definido como "o conhecimento adquirido durante um projeto que mostra como os eventos do projeto foram abordados ou devem ser abordados no futuro, com o objetivo de melhorar o desempenho futuro" (PMI, 2017). Para um maior aprofundamento desta definição, o Dicionário Brasileiro da Língua Portuguesa Michaelis define *evento* como "algo que acontece e que se pode observar" (EVENTO, 2015).

Fica explícito na definição de lições aprendidas a sua relação estreita com o conceito de conhecimento: elas representam um subconjunto do conhecimento adquirido no projeto, mais especificamente aquele que permite melhorar o seu desempenho futuro.

Figura 4 – Relação entre conhecimento e lições aprendidas



Fonte: Elaborado pelo autor

2.4 ANÁLISE COLETIVA DO TRABALHO

A Análise Coletiva do Trabalho (ACT) é um conjunto de técnicas desenvolvidas pela pesquisadora brasileira Leda Leal Ferreira na década de 1990, com o objetivo de gerar em um grupo o entendimento coletivo a respeito de suas próprias práticas de trabalho. Este objetivo vem de encontro à necessidade já discutida do Scrum em socializar-se o conhecimento do projeto, em particular o conhecimento tácito.

O método de trabalho da ACT consiste em provocar o grupo a responder à pergunta "O que você faz no seu trabalho?" da maneira mais exaustiva possível. E nesse processo, o pesquisador incita e encoraja as pessoas a analisarem sua atividade, utilizando-se o que elas próprias têm a dizer sobre o trabalho. (FERREIRA, 2019)

No artigo de Ferreira (2015) é delineado o *modus operandi* desta forma de trabalho em grupo:

- O facilitador estabelece o diálogo inicial com os participantes, mas logo em seguida permite que a dinâmica siga com estes últimos somente;
- O facilitador deve ser um ouvinte e não querer ensinar;
- Não se faz uso de questionários, filmagens ou outros artefatos. Somente a fala dos participantes;
- A última etapa do trabalho é a elaboração de um texto com os resultados da análise, divulgado aos participantes.

A ACT difere de outras técnicas de trabalho em grupo por dois aspectos importantes no reuso de lições aprendidas: 1) Ela fomenta um entendimento coletivo do problema sendo discutido, portanto, equilibra os saberes dentro do grupo e 2) O facilitador atua em segundo plano, deixando que a interação ocorra entre as pessoas, maximizando os processos de externalização e socialização do conhecimento discutidos na seção 2.2.

3. MÉTODO

3.1 REVISÃO SISTEMÁTICA

O método de revisão sistemática (RS) tem suas origens na área da Medicina, e em função de seus resultados positivos avançou tanto em seu campo original quanto sobre outras áreas do conhecimento. A revisão sistemática difere de uma revisão tradicional da literatura principalmente ao adotar um protocolo que, além de reduzir os vieses, permite a auditoria do processo (FELIZARDO *et al.*, 2017). As revisões tradicionais, conduzidas sem um planejamento inicial, têm a sua qualidade muito dependente da experiência dos pesquisadores, podendo ser pouco abrangentes e de difícil repetição, impedindo sua validação por outros pesquisadores. Em Felizardo *et al.* (2017) se esclarece que a revisão sistemática não apenas endereça estas dificuldades, como também apresenta vantagens importantes sobre a pesquisa tradicional:

- A redução de vieses nos resultados da pesquisa;
- A possibilidade de identificar e combinar os dados dos estudos;
- Identificar lacunas da pesquisa atual, sugerindo novas propostas de investigação.

Considerando-se estas vantagens sobre a pesquisa tradicional, bem como a orientação deste trabalho em apresentar evidências sólidas para a construção de um processo

de reuso de lições aprendidas, utilizar-se-á a revisão sistemática da literatura para embasar esse artigo. Seu protocolo é descrito na seção seguinte.

3.2 PROTOCOLO DA REVISÃO SISTEMÁTICA

Uma revisão sistemática inicia-se com a definição de um protocolo de execução, conforme a prática indicada em Felizardo *et al.* (2017), asseverando que "toda RS deve ser realizada de acordo com um plano predefinido". No protocolo são definidos os critérios de decisão que devem ser aplicados aos estudos, a fim de determinar sua inclusão ou exclusão do escopo da pesquisa. Também são definidos os dados a serem extraídos dos estudos, de forma a compor o resultado da revisão.

Detalha-se em seguida o protocolo adotado nesta pesquisa, modelado segundo o padrão descrito em Felizardo *et al.* (2017):

Objetivo: Identificar processos e ferramentas que promovam ou suportem o reuso sistemático de conhecimento por times ágeis, considerando as lições aprendidas em iterações de um mesmo projeto ou em projetos diferentes dentro de uma mesma organização.

Questão de pesquisa: É vital para a pesquisa que a questão esteja bem formulada, considerando-se que esta norteará todos os passos subsequentes. Ainda em Felizardo *et al.* (2017) recomenda-se para tanto o uso dos critérios PICO - estrutura oriunda da pesquisa em medicina - mas que também se aplica à pesquisa em engenharia de software com alguma adaptação de contexto. O Quadro 2 apresenta os critérios PICO e sua aplicação na questão de pesquisa que norteia este artigo:

Como desenvolver um processo que habilite o reuso de lições aprendidas em projetos de desenvolvimento ágil de software?

Quadro 2 - Critérios PICO aplicados à pesquisa

Critério PICO	Descrição do critério	Aplicação neste trabalho
Population (P)	Papel ou grupo que será observado	Times que utilizam métodos ágeis para desenvolvimento de software
Intervention (I)	Objeto da investigação	Reuso de lições aprendidas
Comparison (C)	Objeto ou parâmetro de comparação	Não há dados iniciais para comparação
Outcome (O)	Resultados ou efeitos observados	Processo que habilite o reuso de lições aprendidas

Fonte: Elaborado pelo autor

Palavras-chave: Agile, agile software development, Scrum, Scrumban, XP, Kanban, tool, framework, knowledge management, reuse, lessons learned, retrospective

Idioma: Inglês

String de busca:

(("agile software development" OR "agile" OR "Scrum" OR "ScrumBan" OR "XP" OR "Kanban") AND ("tool" OR "framework" OR "knowledge management") AND "reuse" OR "lessons learned" OR "retrospective"))

Critérios para seleção de fontes: As fontes primárias são bibliotecas digitais indexadas de conteúdos científicos, que possuam as seguintes características:

- Disponibilidade de textos na íntegra
- Que permitam adaptação da string de busca
- Que permitam a exportação dos resultados

Lista das fontes de busca:

- Scopus (https://www.elsevier.com/pt-br/solutions/scopus)
- ACM Digital Library (https://dl.acm.org)
- Web of Science (https://clarivate.com/products/web-of-Science/databases/)
- IEEEXplore (http://ieeexplore.ieee.org/Xplore/)

Estratégia de busca nas fontes:

As buscas serão efetuadas de forma automática nas bases bibliográficas por meio da string de busca. Pequenas adaptações podem ser necessárias para respeitar eventuais particularidades de bases específicas.

Os resultados serão limitados em relação ao ano de publicação, considerando apenas os estudos realizados no período compreendido entre 2017 e 2021, ou seja, abrangendo o período dos cinco últimos anos de publicações.

Critérios para inclusão de estudos:

- I-1) Estudo aborda teoria da gestão do conhecimento aplicável ao contexto de lições aprendidas em projetos ágeis
- I-2) Estudo apresenta processo ou framework para reuso de lições aprendidas

Critérios para exclusão de estudos:

- E-1) Duplicidade
- E-2) Não possui resumo
- E-3) Publicado apenas como um resumo ou pôster
- E-4) Idioma diferente do inglês
- E-5) Não foi possível acessar o conteúdo completo do artigo
- E-6) Não alinhado aos objetivos da pesquisa
- E-7) Não atendeu a nenhum dos critérios de inclusão

Extração de dados

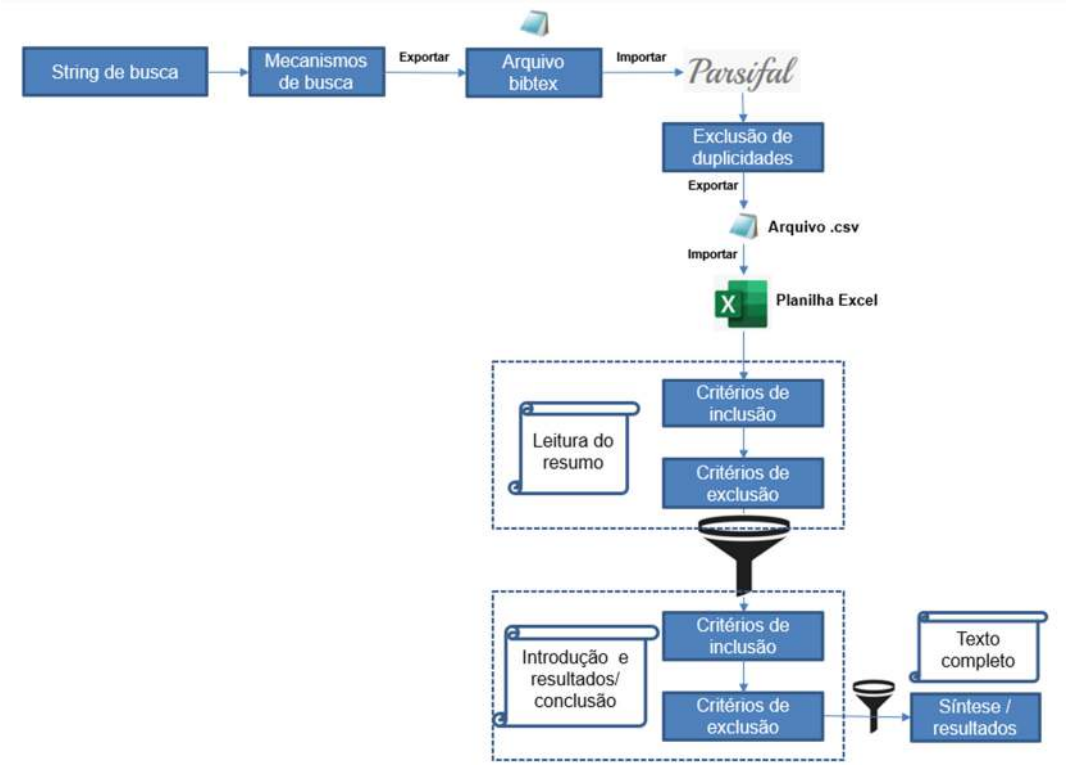
Dos estudos selecionados na fase final, serão extraídas as seguintes informações:

- Aspectos teóricos de gestão do conhecimento que sejam aplicáveis ao reuso de lições aprendidas
- Processos ou *frameworks* que habilitem o reuso de lições aprendidas
- Ferramentas para o registro de conhecimento de projeto

Estratégia para seleção dos estudos:

A Figura 5 exibe o processo de revisão sistemática adotado neste artigo. Suas fases são detalhadas em seguida.

Figura 5 - Processo adotado para a revisão sistemática



Fonte: Elaborado pelo autor

O processo inicia-se com a pesquisa automática nas bases bibliográficas, utilizando as strings de busca documentadas neste protocolo. Os resultados são exportados em arquivos em formato texto no padrão BibTex. Estes arquivos são então importados na ferramenta Parsif.al, para remoção dos estudos duplicados.

Após a exclusão automática de duplicidades, os resultados são exportados para um arquivo em formato csv, e então importados em uma planilha Excel. O uso da planilha auxiliar em Excel se justifica por uma limitação da ferramenta Parsif.al: esta só permite a seleção de estudos em duas etapas. Entretanto, este trabalho prevê três etapas de seleção, tornando-se necessário um sistema auxiliar para a execução da etapa adicional.

Dentro da ferramenta Excel, executa-se a seleção inicial dos estudos aplicando-se os critérios de inclusão e exclusão aos metadados do estudo (título e resumo). Os resultados são registrados na planilha.

Seguindo, serão analisados os capítulos de introdução e conclusão dos estudos selecionados na etapa anterior, reavaliando os critérios de inclusão e exclusão segundo as novas informações obtidas. Os resultados são registrados na planilha. Esta etapa configura o processo intermediário que, não tendo suporte na ferramenta Parsif.al, motivou o uso de uma alternativa.

Por fim, o texto completo dos estudos selecionados na etapa anterior será analisado frente aos critérios de inclusão e exclusão. Esta é etapa final da seleção de estudos, e seu resultado forma o conjunto que será analisado para compor o resultado final do processo e suas conclusões.

4. RESULTADOS

4.1 CONDUÇÃO DA REVISÃO SISTEMÁTICA

A revisão sistemática foi conduzida em dezembro de 2021, retornando um total de 388 artigos das bases primárias. A análise dos artigos selecionados levou à inclusão de 5 artigos adicionais, totalizando 393 estudos. Esta etapa configurou o processo de *pearl growing* admitido pela RS, que pode ser descrito como a etapa onde "o pesquisador deverá utilizar o conhecimento obtido nesses estudos primários para localizar novos estudos primários relevantes." (FELIZARDO et al., 2017).

Na primeira etapa foram excluídos 119 estudos duplicados, restando um total de 274 artigos para avaliação nas fases subsequentes.

Grande parte dos artigos foi excluída durante a fase de análise de metadados (título e resumo), restando apenas 26 estudos relevantes como resultado desta etapa. Em seguida, a aplicação dos critérios de seleção, após a análise de introdução e conclusão, resultou em uma seleção de 14 artigos para leitura completa. Destes, apenas um foi excluído após leitura completa, chegando-se ao conjunto final, que totalizou 13 artigos.

A Tabela 1 apresenta os resultados ora descritos, etapa a etapa, discriminando-os por base de dados:

Tabela 1 - Resultados por base de dados

Etapa	ACM	IEEE	Scopus	WoS	Outras Fontes	Resultado Geral
1) Resultados da busca inicial	24	31	241	92	5	**393**
2) Filtragem com remoção de artigos duplicados	22	24	197	26	5	**274**
3) Seleção pela análise dos metadados	1	4	15	1	5	**26**
4) Seleção pela análise dos capítulos de introdução e conclusão	1	1	6	1	5	**14**
5) Seleção pela análise do texto completo	1	1	5	1	5	**13**

Fonte: Elaborado pelo autor

Os artigos selecionados na etapa final são apresentados em seguida:

Tabela 2 - Artigos selecionados pela RS

Título	Autor	Ano	Fonte
Synthesizing researches on Knowledge Management and Agile Software Development using the Meta-ethnography method	Napoleão *et al*.	2021	Scopus
Knowledge management in the software industry: how Scrum activities	Tenorio *et al*.	2020	WoS

Título	Autor	Ano	Fonte
support a knowledge management cycle			
Playing with Your Project Data in Scrum Retrospectives	Matthies, C.	2020	Scopus
Adapting Scrum Process with 7C Knowledge Management Model	Gandomani *et al*.	2019	IEEE
Implementing knowledge management in agile projects by pragmatic modeling	Störrle, H.	2018	Scopus
Experiential Team Learning in Software Startups	Khanna, D.	2018	ACM
Knowledge transfer in a management process for outsourced agile software development	Brito *et al*.	2017	Scopus
Knowledge management and reflective practice in daily stand-up and retrospective meetings	Andriyani, Y.	2017	Scopus
Análise Coletiva do Trabalho: quer ver? Escuta	Ferreira, L.	2015	Outras fontes
An ontological approach for Program Management Lessons Learned: Case study at Motorola Penang Design Centre	Cheah *et al*.	2011	Outras fontes
Self Adaptability of Agile Software Processes: A Case Study on Post-iteration Workshops	Salo *et al*.	2004	Outras fontes
The 7C Model for Organizational Knowledge Creation and Management	Oinas-Kukkonen, H.	2004	Outras fontes
Extending Agile Methods: Postmortem Reviews as Extended Feedback	Dingsøyr, T.; Hanssen, G.	2003	Outras fontes

Fonte: Elaborado pelo autor

A seção seguinte apresenta a análise dos resultados extraídos destes estudos.

4.2 ANÁLISE DOS RESULTADOS

Em Napoleão *et al.* (2021) os autores apontam a dificuldade em reutilizar o conhecimento como uma causa de dificuldades em projetos, dos quais o sucesso depende do conhecimento e experiência dos desenvolvedores. Também esclarecem, justificando a condução desta pesquisa, que a maior parte do conhecimento gerado nas organizações não é processado, sendo difícil de ser articulado e, portanto, reutilizado. De forma mais específica, os autores apresentam como resultado relevante o diagnóstico que a Retrospectiva é ineficiente no reuso das lições aprendidas quando não documenta o conhecimento e não monitora a aplicação do que foi gerado durante a cerimônia.

No trabalho de Tenório *et al*. (2020) revela-se que, segundo o *Europe Committee for Standardization*, há mais de 150 de modelos de gestão de conhecimento, dando uma idéia da complexidade do problema do reuso. Nesse mesmo trabalho, faz-se um paralelo entre o ciclo

do *framework* Scrum e um ciclo de gestão de conhecimento, chegando-se à conclusão de que há um alinhamento entre o Scrum e um ciclo de conhecimento formado pelas etapas de captura, criação, armazenamento, disseminação, compartilhamento e uso de conhecimento. Dentro deste ciclo, a Retrospectiva é o momento no qual estão presentes as etapas de captura, criação, disseminação e compartilhamento do conhecimento que, em última instância, poderá ser reutilizado para melhoria futura do projeto.

Já em um trabalho mais focado na Retrospectiva, Matthies (2020) observa que é comum que times Scrum utilizem formatos diversos de Retrospectiva, e que na literatura cinzenta existem mais de 30 tipos dos chamados "*Retrospective Games*", que priorizam apenas as percepções e impressões subjetivas do time, enviesando seus resultados. Como alternativa, o autor apresenta o conceito de mineração de repositório com o objetivo de coletar dados que servirão de *input* para a cerimônia (por exemplo: *code reviews,* defeitos, métricas da Sprint, *logs* e outros relatórios). Esta prática se relaciona ao reuso de lições aprendidas no sentido em que evidencia de maneira inequívoca como os eventos do projeto foram abordados.

Christoph Matthies (2020) observa que é comum que times Scrum utilizem diversos formatos de Retrospectiva. Observa também que na literatura cinzenta existem mais de 30 tipos de "Retrospective Games", que priorizam as percepções e impressões subjetivas do time, em detrimento dos fatos objetivos ocorridos na Sprint, trazendo viés aos resultados da reflexão

Gandomani *et al.* (2019) apresenta o modelo de gestão de conhecimento 7C, cuja importância é delimitar um conjunto de habilitadores que, ao serem integrados, levam à criação de conhecimento em forma de uma memória organizacional. Ao adaptar o *framework* Scrum ao modelo de 7C, enquadra a Retrospectiva como a atividade que suporta o reuso de conhecimento, contribuindo para a melhoria do projeto.

Em uma outra perspectiva do problema, Harald Störrle (2018) apresenta o conceito de modelamento pragmático, com a proposta que uma documentação boa o suficiente pode ser obtida através dos chamados modelóides e diagramas, que são modelos "menos próprios", mas que registram o conhecimento capturado de forma relevante. Importante também salientar a sugestão do autor em utilizar um sistema comum de publicação (wiki) como repositório, destacando sua funcionalidade de busca de texto completo.

O artigo de Dron Khanna (2018) informa que a Retrospectiva habilita o reuso do conhecimento que melhora a performance de projetos na organização, no sentido em que ela é uma "oportunidade para os desenvolvedores entenderem melhor o processo de desenvolvimento de software" (KHANNA, 2018). Este resultado se alinha com a definição utilizada aqui de lições aprendidas. O autor também esclarece que, dentre as diversas práticas utilizadas para se conduzir uma Retrospectiva, o Post Iteration Workshop (PIW) e Postmortem Analysis (PMA) são as mais populares.

Contribuindo para o entendimento do papel da Retrospectiva no reuso de lições aprendidas, a pesquisa de De Brito *et al*. (2017) a coloca explicitamente como a oportunidade onde se discutem, capturam e assimilam lições aprendidas. Este trabalho também sugere o uso de uma *wiki* como repositório de conhecimento.

Encerrando a revisão individual dos artigos, o estudo de Yanti Andriyani (2017) traz como resultado relevante para a pesquisa o entendimento dos três tipos de conhecimento utilizados em engenharia de software (projeto, processo e produto), dos quais se podem derivar lições aprendidas. Em um outro aspecto, este artigo também cita o PIW e PMA como práticas utilizadas para conduzir Retrospectivas, confirmando o resultado discutido em (KHANNA, 2018).

O conhecimento adquirido com a leitura destes textos levou à inclusão de outras referências na seleção, a saber: os estudos de Salo et al. (2004) e Dingsøyr e Hanssen (2003) que detalham, respectivamente, como as práticas de Post-Iteration Workshop e Lightweight Postmortem Analysis foram concebidas como modelos de Retrospectiva. O texto de Oinas-Kukkonen (2004), que define aspectos do processo 7C que não estavam suficientemente esclarecidos no texto de Gandomani et al. (2019), que havia sido selecionado na fase anterior. Também relevante ao processo proposto são as técnicas de Análise Coletiva do Trabalho (ACT) elicitadas no texto de Leda Leal Ferreira (2015). E por fim, o estudo de caso relatado em Cheah et al. (2011) que propõe uma ontologia para documentar lições aprendidas.

Há dois resultados importantes retirados dos artigos selecionados. O primeiro deles é o consenso sobre a cerimônia de Retrospectiva como a prática ágil que habilita o reuso de lições aprendidas, no sentido em que esta compartilha e dissemina as melhores práticas do projeto, orientando a maneira como eventos futuros devem ser abordados na direção de melhorá-lo.

O segundo resultado é que esta cerimônia, em suas formas mais comuns, apresenta problemas que inibem a reutilização de lições aprendidas, ao impedir a consolidação ou disseminação de conhecimento que poderia ser utilizado para melhoria futura do projeto. Os artigos selecionados revelam a existência de cinco problemas particularmente importantes para o tema, que ocorrem nas formas mais populares de Retrospectiva:

I) A priorização de percepções subjetivas dos participantes em detrimento de fatos objetivos

Mathies (2020) cita um estudo conduzido na Microsoft onde se conclui que “as crenças dos programadores são baseadas primeiramente na experiência pessoal, ao invés das evidências empíricas do projeto” (MATTHIES, 2020). Este fato impacta as lições aprendidas no sentido em que a retrospectiva tende a basear-se apenas em conhecimento tácito, possivelmente em fatos que não correspondem a realidade, enviesando qualquer tipo de aprendizado que poderia ocorrer nesta cerimônia.

II) Não inspecionar os artefatos gerados durante a Sprint

Parte do conhecimento explícito do projeto, armazenado nos repositórios, não é revisto. Esse conhecimento é fundamental para a melhoria futura do projeto, no sentido em que a medição objetiva destes artefatos, como ação de retrospectivas, são um caminho para quantificar os resultados das melhorias propostas (Matthies, 2020).

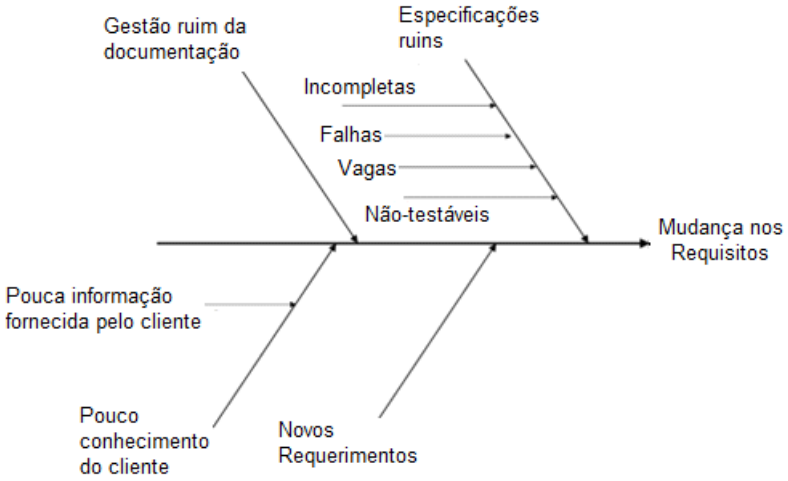
III) Despriorização ou abandono da análise de causa raiz dos problemas encontrados

Khanna (2018) esclarece que o *Post Iteration Workshop* (PIW) é uma das técnicas populares para a condução de retrospectivas. Os idealizadores do PIW reconhecem terem simplificado a análise de causa raiz, como está documentado no trabalho de Salo *et al.* (2004): “...uma técnica de análise de causa raiz chamada diagrama Ishikawa, para análise de causas subjacentes, [...], foi considerada, porém não incluída”.

O diagrama de Ishikawa exibe em um formato compacto as diversas causas subjacentes que podem levar ao problema que é objeto de estudo. Este diagrama já havia sido utilizado em outras técnicas de retrospectiva (DINGSØYR; HANSSEN, 2003). A

Figura ura 6 mostra um diagrama de Ishikawa:

Figura 6 – Diagrama de Ishikawa para “mudança nos requisitos”



Fonte: Adaptado de Dingsøyr e Hanssen (2003)

Matthies (2020) menciona o site *Retromat* (BALDAUF, 2018) como uma referência popular para o planejamento de retrospectivas. Neste site é possível encontrar 22 técnicas sugeridas para a geração de *insights* sobre os problemas levantados na retrospectiva, fase na qual se dá a discussão e análise de causas subjacentes. Dessas 22 técnicas, apenas uma sugere a utilização de um diagrama de causa e efeito, demonstrando a baixa popularidade desta prática tão relevante ao processo de aprendizado em projetos

IV) Falta de documentação

Através de entrevistas com especialistas em desenvolvimento ágil de software, um estudo publicado em 2021 no Journal of Systems & Software diagnosticou que “devido à falta de documentação e monitoramento da aplicação do conhecimento gerado nesta reunião, a melhoria raramente acontece” (NAPOLEÃO *et al.*, 2021). Novamente, estabelece-se um impacto no reaproveitamento de lições aprendidas, no sentido em que a falta de documentação evita que o conhecimento gerado seja utilizado para melhoria futura do projeto, e a falta de monitoramento sobre o resultado das ações propostas desacelera a sistematização do processo de melhoria do projeto.

V) Falta de monitoramento da aplicação do conhecimento gerado

Como citado no item anterior, em Napoleão *et al.* (2021) assevera-se que que “devido à falta de documentação e monitoramento da aplicação do conhecimento gerado nesta reunião, a melhoria raramente acontece”. Ao impedir a melhoria futura do projeto, fica claro que este problema se traduz em ineficácia no processo de reutilização das lições aprendidas no projeto.

Em resumo, da análise destes resultados decorre que a Retrospectiva é um momento chave, dentro do framework ágil, para disseminação e consequente reuso das lições aprendidas no projeto. Entretanto, as técnicas popularmente utilizadas para esta prática

apresentam problemas que impactam de alguma forma o processo de aprendizado do time. O

Quadro 3 sumariza estes problemas e seu impacto no reuso de lições aprendidas:

Quadro 3 – Problemas comuns identificados na condução de Retrospectivas

	Problema	Como impacta o reuso de lições aprendidas	Referências
I	A priorização de percepções subjetivas dos participantes em detrimento de fatos objetivos	Enviesando o entendimento sobre a maneira pela qual os **eventos do projeto** foram abordados.	(MATTHIES, 2020)
II	Não inspecionar os artefatos gerados durante a Sprint	Não avalia a evolução dos artefatos em função das ações de **melhoria de desempenho** propostas em retrospectivas anteriores.	(KHANNA, 2018), (MATTHIES, 2020)
III	Despriorização ou abandono da análise de causa raiz dos problemas encontrados	Ignora conhecimento que poderia ser utilizado na **melhoria do desempenho futuro** do projeto.	(SALO *et al.*, 2004)
IV	Falta de documentação	Não registra o conhecimento explícito que poderia ser utilizado **na melhoria do desempenho futuro** do projeto	(NAPOLEÃO *et al.*, 2021)
V	Falta de monitoramento da aplicação do conhecimento gerado	Não assegura que o conhecimento gerado está sendo utilizado na **melhoria do desempenho** do projeto	(NAPOLEÃO *et al.*, 2021)

Fonte: Elaborado pelo autor

No capítulo seguinte é apresentada uma proposta para mitigar estes problemas, com o objetivo final de tornar a Retrospectiva uma prática que fomente o reuso das lições aprendidas no projeto, de uma forma mais eficaz.

4.3 PROCESSO PROPOSTO

O objetivo desta seção é construir um processo de Retrospectiva que habilite o reuso de lições aprendidas, com base nos elementos encontrados na literatura.

4.3.1 Retrospectiva de Sprint

A estratégia para construir um processo otimizado de Retrospectiva é incorporar técnicas que mitiguem os problemas identificados na literatura, já descritos anteriormente neste artigo e resumidos no

Quadro 3.

A literatura também revela técnicas que podem mitigar estes problemas. O Quadro 4 exibe, para cada um dos problemas identificados, as técnicas utilizadas e os artigos que as suportam. O detalhamento de cada uma delas se dará nas seções seguintes.

Quadro 4 – Técnicas propostas para mitigar os problemas em Retrospectivas

	Problema	Técnica(s) proposta(s)	Referências
I	A priorização de percepções subjetivas dos participantes em detrimento de fatos objetivos	Utilizar os dados do repositório para orientar a Retrospectiva	(MATTHIES, 2020)
II	Não inspecionar os artefatos gerados durante a Sprint	Utilizar os próprios artefatos da Sprint para orientar a Retrospectiva	(MATTHIES, 2020)
III	Despriorização ou abandono da análise de causa raiz dos problemas encontrados	Uso de diagramas de Ishikawa para enumerar causas subjacentes aos problemas	(DINGSØYR; HANSSEN, 2003)
IV	Falta de documentação	Criação colaborativa de esboços e modelóides (modelamento pragmático); Utilização de uma ontologia para armazenar lições aprendidas	(CHEAH *et al.*, 2011) (STÖRRLE, 2018)
V	Falta de monitoramento da aplicação do conhecimento gerado	Revisar as ações do workshop anterior como parte integrante do processo	(SALO *et al.*, 2004)

Fonte: Elaborado pelo autor

Adota-se como base para a Retrospectiva a prática do Post Iteration Workshop (PIW), por duas razões:

1) É uma das técnicas mais comuns utilizadas por times ágeis, como destacado por (KHANNA, 2018) e (ANDRIYANI, 2017);

2) Sua estrutura é compatível com a prática prescrita pelo Scrum Guide (SCHWABER; SUTHERLAND, 2020).

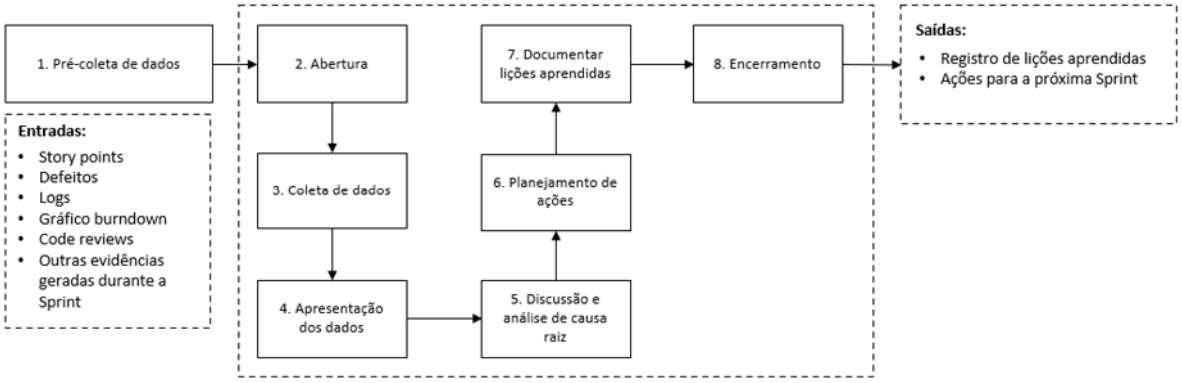
O PIW é descrito no trabalho de Salo *et al.* (2004), no qual se elenca as suas etapas distintas:

- Coletar e estruturar dados;
- Discutir os dados;
- Coletar sugestões para melhoria;
- Determinar as ações para a próxima Sprint;
- Revisar ações do último workshop.

Ao incorporar a revisão das ações passadas dentro da própria cerimônia, esta prática mitiga a falta de monitoramento das ações (problema V), apontado em Napoleão *et al.* (2021).

Ao se incorporar as demais técnicas apresentadas no Quadro 4, tem-se o processo aqui proposto para a Retrospectiva de Sprint:

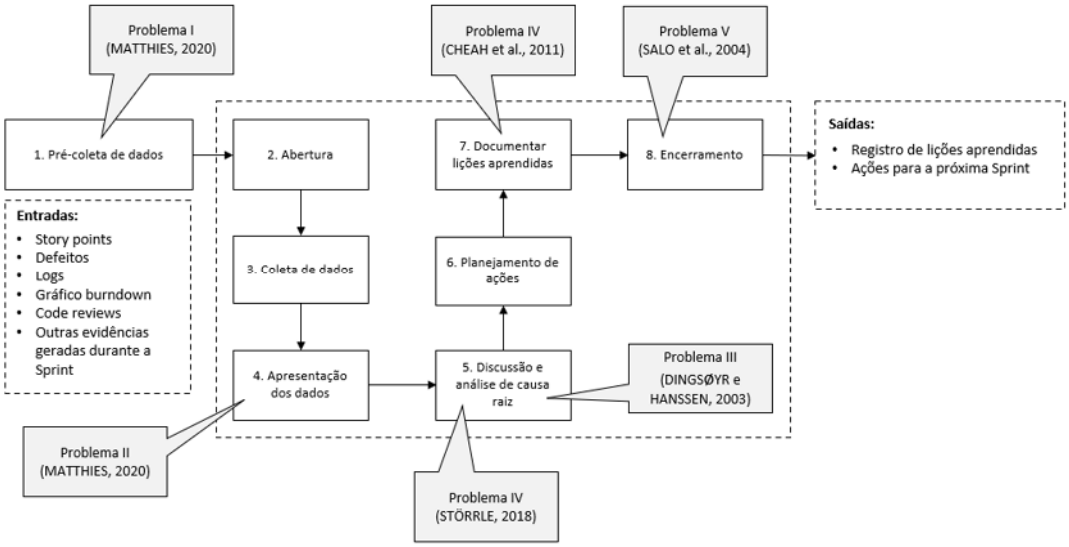
Figura 7 – Processo proposto para a Retrospectiva



Fonte: Elaborado pelo autor

A Figura 8 exibe os pontos do processo em que foram incorporados os elementos de mitigação encontrados nos artigos:

Figura 8 – Artigos de suporte ao processo proposto



Fonte: Elaborado pelo autor

As seções seguintes detalham cada etapa do processo.

4.3.2 Pré-coleta de dados

Matthies (2020) introduz o conceito de retrospectiva orientada a dados, na qual se utilizam dados e informações oriundas dos diversos repositórios utilizados pelo time durante a execução da Sprint. Estas informações são usadas como entrada para a fase de coleta de dados. Ao utilizar esta técnica, o time é confrontado com a realidade do que ocorreu objetivamente na Sprint e o resultado de suas ações. Nesse cenário, é então colocado um contraponto objetivo a qualquer percepção ou interpretação subjetiva que seja exposta durante a cerimônia. Esta prática mitiga o problema da subjetividade que permeia a cerimônia.

O Quadro 5 exibe alguns exemplos de dados que podem ser extraídos das diversas ferramentas de suporte ao desenvolvimento. Estes configuram o material a ser coletado previamente para servir de base ao processo de retrospectiva:

Quadro 5 – Exemplos de dados gerados durante uma Sprint

Ferramental	Exemplos de dados que podem ser extraídos
Controle de versão	Fragmentos de código, detalhes dos *commits, eventos*
Gerenciamento de atividades	Tarefas atribuídas, atualizações de status
Teste de software	Logs de testes, informações de builds
Monitores de status	Disponibilidade da aplicação, eventos de indisponibilidade
Revisão de código	Tempo de finalização, comentários dos revisores, aprovações
Análise de código	Resultados de cobertura de código, verificação de estilo de código

Fonte: Adaptado de (MATTHIES, 2020)

Tornar a retrospectiva orientada a dados possui um segundo efeito benéfico: realizar a inspeção dos artefatos gerados na Sprint. Dessa forma, executa-se a prescrição do Scrum Guide que é, de fato, inspecionar a Sprint, e cuja inobservância também configura um problema comum desta cerimônia, como já apontado anteriormente.

4.3.2 Abertura

É sintomático que os textos encontrados na revisão da literatura descrevam técnicas para a condução de retrospectivas sem mencionar seu processo de abertura. Aqui, as técnicas de ACT vêm contribuir para a experiência, esclarecendo que, nestes processos, devem existir etapas preliminares às reuniões de fato, cabendo ao pesquisador "explicar aos participantes os objetivos do projeto e as regras das reuniões" (FERREIRA, 2019). Assim, na abertura da Retrospectiva o facilitador deve fazer uma breve introdução de como se dará a cerimônia, nivelando o entendimento do grupo sobre o funcionamento do processo e seus objetivos.

4.3.3 Coleta de dados

Esta etapa está presente nas formas mais comuns de Retrospectiva e consiste em extrair do time as suas percepções sobre o trabalho efetuado na iteração anterior. Ainda que este processo incorpore dados objetivos da Sprint conforme proposto por Matthies (2020), a leitura das percepções do time é importante para confrontá-las com a realidade mostrada pelos dados. Além disso, o Scrum Guide prescreve que na Retrospectiva sejam inspecionados "indivíduos e interações" (SCHWABER; SUTHERLAND, 2020) e, para este fim, é necessário também coletar as percepções subjetivas. Isto é feito através de três perguntas direcionadas ao time:

- O que correu bem durante a Sprint?
- O que não foi bem durante a Sprint?
- O que podemos melhorar?

Perguntas como estas são largamente utilizadas em Retrospectivas (KHANNA, 2018), e têm por objetivo extrair as percepções subjetivas dos participantes, complementando os dados obtidos na fase de pré-coleta.

4.3.4 Apresentação dos dados

A etapa de apresentação dos dados deve prover clareza dos temas em dicussão, e tornar explícitos tanto os dados coletados na fase de pré-coleta, quanto os temas apontados pela equipe durante a fase de coleta. Para a primeira, recomenda-se a utilização do próprio repositório de dados (por exemplo, Jira) para a apresentação de um relatório de Sprint. Para os demais, a literatura sugere o uso de boards e post-its (DINGSØYR; HANSSEN, 2003). Implementações em software destes insumos podem ser facilmente encontrados na internet.

4.3.5 Discussão e análise de causa-raiz

Como observado por Matthies (2020), a análise de causa raiz não aparece nas técnicas tradicionais de Retrospectiva, a despeito de sua prescrição no Scrum Guide (SCHWABER; SUTHERLAND, 2020). Entretanto, esta prática foi utilizada anteriormente na técnica *Lightweight Postmortem Review* por meio do uso de diagramas de Ishikawa (DINGSØYR; HANSSEN, 2003).

Para cada problema relevante encontrado durante a cerimônia, um diagrama deve ser construído colaborativamente. Assim como no item anterior, um quadro analógico ou digital deve ser utilizado. O desenvolvimento colaborativo que permeia esta etapa materializa os subprocessos de conceitualização e colaboração do modelo 7C.

A literatura pouco fala sobre o papel do facilitador da cerimônia. Nesse sentido, apresenta-se uma oportunidade de adicionar elementos que permitam melhorar o processo. Aqui, as técnicas da Análise Coletiva do Trabalho (ACT) são de grande valia como catalisadoras do aprendizado em grupo. Particularmente, duas estratégias da ACT são úteis neste ponto do processo:

- Todos os participantes devem entender o problema analisado. Para isto devem fazer perguntas e ajudar a respondê-las;
- Um dos participantes (possivelmente quem teve maior interação com o problema durante a Sprint) deve explicar ao grupo o problema de forma mais detalhada.

O objetivo do trabalho é tornar claro o entendimento, pelo grupo, do problema em discussão. Para tanto, a ACT esclarece ao facilitador que este deve iniciar o diálogo com a equipe e logo fomentar a discussão entre os participantes, quase como que se retirando do processo, ou melhor, sendo um observador deste. Não se deve assumir a tarefa de ensinar ou impor uma solução, mas sim deixar que a equipe chegue ao entendimento. Esta dinâmica é conduzida pela pergunta chave da ACT: “Como você faz o seu trabalho?”, que deve ser adaptada para o contexto da discussão, e pode assumir duas funções distintas:

- Ao discutir o que **correu bem** durante a Sprint, a pergunta “Como você faz seu trabalho?” permite que os participantes mais experientes esclareçam ao grupo como as boas práticas foram implementadas, servindo assim ao propósito de socializar o conhecimento com os participantes menos experientes do grupo;
- Ao discutir o que **não correu bem** durante a Sprint, a mesma pergunta assume um caráter investigativo, permitindo que os participantes mais envolvidos com o problema em discussão descrevam os passos que levaram ao insucesso. Isto servirá de fio condutor para que o grupo descubra as relações de causa e efeito que se manifestaram e, assim, gerar lições aprendidas.

4.3.6 Planejamento de ações

O PIW, técnica base utilizada para a construção deste processo, incorpora em sua etapa final uma discussão sobre as ações a serem tomadas na próxima Sprint visando a melhoria do processo (SALO *et al.*, 2004). Nesta etapa também são revisadas as ações discutidas na Sprint anterior, com o objetivo de monitorar sua implementação, e assim governar a evolução do processo.

4.3.7 Documentar as lições aprendidas

O registro deve ser feito em uma *wiki* com uma ontologia adequada para o armazenamento de lições aprendidas, como sugerido em Cheah *et al*. (2011):

Quadro 6 - Ontologia para armazenamento de lições aprendidas

Elemento	Descrição
Título	Um texto que descreva sucintamente a lição documentada
Descrição / Observação	Descrição da situação ou da observação de um evento positivo ou negativo do projeto, que é o objeto da lição documentada. Aqui podem ser armazenados os modelos pragmáticos que se apliquem
Impacto	Informa como o evento impactou o projeto (custo / prazo / qualidade / outros)
Ação	Descreve a resolução do evento, caso exista. Aplica-se em especial aos eventos negativos

Fonte: Adaptado de Cheah et al. (2011)

Os estudos de De Brito *et al.* (2017) e Störrle (2018) sugerem o uso de wikis como repositório de informações. Esta ferramenta possui como vantagens sua facilidade de uso, sua versatilidade para o armazenamento de diferentes tipos de dados (tabelas, texto, imagens) e a possibilidade de executar buscas de texto completo.

A wiki deve ser configurada para permitir o armazenamento dos itens descritos na ontologia. Um outro aspecto importante a se destacar nesse ponto é o uso do modelamento pragmático introduzido em Störrle (2018), que consiste em se aperfeiçoar artefatos documentais (esboços e modelóides), tornando-os minimamente aptos a serem utilizados como documentação. Este esboços e modelóides confeccionados durante a Retrospectiva ou recuperados pela equipe devem ser armazenados como parte das lições aprendidas.

4.3.8 Encerramento

Assim como na abertura (seção 4.3.2), a revisão da literatura tampouco revela detalhes sobre o processo de encerramento de uma Retrospectiva de Sprint. Novamente, as técnicas de ACT vêm enriquecer o processo. Como Ferreira (2015) aponta, “a última etapa de uma ACT é a elaboração e divulgação de um texto dando conta dos resultados e análises”

Considerando o objetivo principal de detectar erros de interpretação e prover esclarecimentos finais, o encerramento da Retrospectiva é uma oportunidade para o facilitador revisar os pontos principais do trabalho efetuado durante a cerimônia, e fazê-lo de forma escrita é uma oportunidade de “socializar tudo o que aprendemos sobre o(s) trabalho(s) que estamos analisando” (FERREIRA, 2015). Assim, a Retrospectiva termina com um breve sumário dos pontos discutidos, de forma escrita, que é armazenado na wiki e divulgado aos participantes.

5. CONCLUSÃO

Este artigo responde à pergunta de pesquisa "como desenvolver um processo que habilite o reuso de lições aprendidas em projetos de desenvolvimento ágil de software?" através de uma revisão sistemática da literatura, que revelou dois resultados seminais sobre o conceito de lições aprendidas em projetos Scrum: 1) que a Retrospectiva é a principal cerimônia onde se as lições são disseminadas e 2) que a mesma Retrospectiva em suas formas mais populares apresenta problemas que inibem o processo de reuso. A literatura também revela algumas práticas que podem ser utilizadas para mitigar estes problemas. Estas práticas foram aqui sintetizadas em um processo proposto para a condução de uma Retrospectiva que habilite o reuso de lições aprendidas pelo time.

A revisão sistemática efetuada e o processo proposto são contribuições deste artigo, que podem auxiliar as organizações a fomentar o reuso de lições aprendidas em projetos Scrum.

São limitações deste artigo, além do processo não ter sido exercitado em um projeto de fato, a dificuldade em medir sua eficácia. O reuso de lições aprendidas pode ser um indicador de difícil implementação prática. Seus efeitos podem requerer um tempo longo para sensibilizar as métricas impactadas.

E considerando-se os pontos expostos, recomendam-se como trabalhos futuros a implementação do processo em um projeto real, e a investigação da relação existente entre as demais cerimônias do Scrum com o reuso de lições aprendidas.

6. REFERÊNCIAS

ADNAN, M.; AFZAL, M. Ontology Based Multiagent Effort Estimation System for Scrum Agile Method. **IEEE Access**, v. 5, p. 25993–26005, 2017.

ANDRIYANI, Y. Knowledge Management and Reflective Practice in Daily Stand-Up and Retrospective Meetings. Em: BAUMEISTER, H.; LICHTER, H.; RIEBISCH, M. (Eds.). **Agile Processes in Software Engineering and Extreme Programming**. Lecture Notes in Business Information Processing. Cham: Springer International Publishing, 2017. v. 283p. 285–291.

BALDAUF, Corinna. **Retromat**. 2018. Disponível em: https://retromat.org/. Acesso em: 04 jun. 2022.

BECK, Kent *et al*. **Manifesto para Desenvolvimento Ágil de Software**. 2001. Disponível em: https://agilemanifesto.org/iso/ptbr/manifesto.html. Acesso em: 04 jun. 2022.

CHEAH, Y.-N.; KHOH, S. B.; OOI, G. B. **An ontological approach for Program Management Lessons Learned: Case study at Motorola Penang Design Centre**. 2011 IEEE International Conference on Industrial Engineering and Engineering Management. **Anais**... Em: 2011 IEEE INTERNATIONAL CONFERENCE ON INDUSTRIAL ENGINEERING AND ENGINEERING MANAGEMENT (IEEM). Singapore, Singapore: IEEE, dez. 2011. Disponível em: <http://ieeexplore.ieee.org/document/6118189/>. Acesso em: 04 jun. 2022.

CISCO. **Cisco Survey Reveals Close to Three-Fourths of IoT Projects Are Failing**. 2017. Disponível em: https://newsroom.cisco.com/c/r/newsroom/en/us/a/y2017/m05/cisco-survey-reveals-close-to-three-fourths-of-iot-projects-are-failing.html. Acesso em: 04 jun. 2022.

CISCO. **The Journey to IoT Value**: challenges, breakthroughs, and best practices. Challenges, Breakthroughs, and Best Practices. 2017. Disponível em: https://www.slideshare.net/CiscoBusinessInsights/journey-to-iot-value-76163389. Acesso em: 04 jun. 2022.

COMPLEX, Simple. **Parsifal**: perform systematic literature reviews. Perform Systematic Literature Reviews. 2021. Disponível em: https://parsif.al/. Acesso em: 04 jun. 2022.

DARFEUILLE, C. P. V. **How to systematically learn lessons from projects: The paradigm of project ReviewsVira Liubchenko**. 2017 12th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT). **Anais**...set. 2017.

DE BRITO, M. F. et al. **Knowledge Transfer in a Management Process for Outsourced Agile Software Development**. Em: HAWAII INTERNATIONAL CONFERENCE ON SYSTEM SCIENCES. 2017. Disponível em: <http://hdl.handle.net/10125/41921>. Acesso em: 04 jun. 2022.

DINGSØYR, Torgeir; HANSSEN, Geir Kjetil. Extending Agile Methods: Postmortem Reviews as Extended Feedback. In: **Advances in learning software organizations: 4th International Workshop, LSO 2002, Chicago, IL, USA, August 6, 2002: revised papers**. Berlin: New York: Springer, 2003.

EVENTO. In: MICHAELIS, Dicionário Brasileiro da Língua Portuguesa. São Paulo: Melhoramentos, 2022. Disponível em: <https://michaelis.uol.com.br/moderno-portugues/busca/portugues-brasileiro/evento/>. Acesso em: 04 jun. 2022.

FELIZARDO, Katia Romero; NAKAGAWA, Elisa Yumi; FABBRI, Sandra Camargo Pinto Ferraz; FERRARI, Fabiano Cutigi. **Revisão sistemática da literatura em engenharia de software**: teoria e prática. Rio de Janeiro: Elsevier, 2017

FERREIRA, Leda Leal. Análise Coletiva do Trabalho: Quer ver? Escuta. **Revista Ciências do Trabalho**, [*s. l*], v. 1, n. 4, p. 125-137, jun. 2015.

FERREIRA, Leda Leal. Lições de professores sobre suas alegrias e dores no trabalho. **Cadernos de Saúde Pública**, Rio de Janeiro, v. 35, n. 13, suppl. 13, p. 1-11, dez. 2019.

GANDOMANI, T. J. et al. **Adapting Scrum Process with 7C Knowledge Management Model**. 2019 5th Conference on Knowledge Based Engineering and Innovation (KBEI). **Anais**... Em: 2019 5TH CONFERENCE ON KNOWLEDGE BASED ENGINEERING AND INNOVATION (KBEI). Tehran, Iran: IEEE, fev. 2019. Disponível em: <https://ieeexplore.ieee.org/document/8735008/>. Acesso em: 04 jun. 2022.

INDUMINI, U.; VASANTHAPRIYAN, S. **Knowledge Management in Agile Software Development- A Literature Review**. 2018 National Information Technology Conference (NITC). **Anais**... Em: 2018 NATIONAL INFORMATION TECHNOLOGY CONFERENCE (NITC). Colombo: IEEE, out. 2018. Disponível em: <https://ieeexplore.ieee.org/document/8550066/>. Acesso em: 04 jun. 2022.

KHANNA, D. **Experiential team learning in software startups**. Proceedings of the 19th International Conference on Agile Software Development: Companion. **Anais**... Em: XP '18 COMPANION: 19TH INTERNATIONAL CONFERENCE ON AGILE SOFTWARE DEVELOPMENT. Porto Portugal: ACM, 21 maio 2018. Disponível em: <https://dl.acm.org/doi/10.1145/3234152.3314992>. Acesso em: 04 jun. 2022.

MATTHIES, C. **Playing with your project data in scrum retrospectives**. Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Companion Proceedings. **Anais**... Em: ICSE '20: 42ND INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING. Seoul South Korea: ACM, 27 jun. 2020. Disponível em: <https://dl.acm.org/doi/10.1145/3377812.3382164>. Acesso em: 04 jun. 2022.

NAPOLEÃO, B. M. et al. Synthesizing researches on Knowledge Management and Agile Software Development using the Meta-ethnography method. **Journal of Systems and Software**, v. 178, p. 110973, ago. 2021.

PMI. **Um Guia do Conhecimento em Gerenciamento de Projetos (Guia PMBOK)**. 6. ed. Newtown Square: Project Management Institute, 2017. 726 p.

SALO, O. *et al.* Self-Adaptability of Agile Software Processes: A Case Study on Post-iteration Workshops. Em: ECKSTEIN, J.; BAUMEISTER, H. (Eds.); **Extreme programming and agile processes in software engineering: 5th international conference, XP 2004, Garmisch-Partenkirchen, Germany, June 6-10, 2004: proceedings**. Berlin; Hong Kong: Springer-Verlag, 2004.

SCHWABER, Ken; SUTHERLAND, Jeff. **The Scrum Guide**: the definitive guide to Scrum: the rules of the game. [S. L.]: Scrum.org, 2020.

ŠIMKONIS, Saulius; SKYRIUS, Rimvydas. Lessons learned: definition-related issues. **Verslo Ir Teisės Aktualijos**: Current Issues of Business and Law, [S. L.], v. 1, n. 8, p. 120-135, ago. 2013.

STÖRRLE, H. Implementing Knowledge Management in Agile Projects by Pragmatic Modeling. Em: SCHAEFER, I.; KARAGIANNIS, D.; VOGELSANG, A.; MÉNDEZ, D.; SEIDL, C. (Eds.): Modellierung 2018, **Lecture Notes in Informatics (LNI),** Gesellschaft für Informatik, Bonn 2018.

STANDISH GROUP. **Chaos Report 2015**. [S. L.]: The Standish Group International, 2015. Disponível em: https://www.standishgroup.com/sample_research_files/CHAOSReport2015-Final.pdf. Acesso em: 04 jun. 2022.

TENÓRIO, N. et al. Knowledge management in the software industry: how Scrum activities support a knowledge management cycle. **Navus - Revista de Gestão e Tecnologia**, v. 10, p. 01–13, 1 jan. 2020.

WILLIAMS, T. How Do Organizations Learn Lessons From Projects—And Do They? **IEEE Transactions on Engineering Management**, v. 55, n. 2, p. 248–266, maio 2008.