Propondo uma abordagem ágil para analisar o impacto sistêmico do software em Engenharia de Requisitos

Mariana Lana Sales[1], **Hugo Bastos de Paula**[1]

[1]Bacharelado em Engenharia de Software
Instituto de Ciências Exatas e Informática - PUC Minas
Rua Cláudio Manoel, 1.162, Funcionários, Belo Horizonte – MG – Brasil

mariana.sales.966782@sga.pucminas.br, hugo@pucminas.br

Abstract. *Currently, Requirements Engineering (RE) support the definition of clear and complete requirements, enabling the delivery of suitable and really needed functionalities. However, there is a set of socio-economic aspects and values that are rarely considered. Due to the lack of systemic impact analysis, some softwares cause unintended consequences in society, from jobs unfeasibility to racial segregation. For this reason, we propose an agile impact analysis framework as part of RE. The objective is visualize systemic impacts that the software may have, so that engineers have the opportunity to change the predicted scenario through adjustments in the requirements. Systemic approaches previously proposed were evaluated and integrated during the framework construction. Then it was evaluated by students and the results show effectiveness in raising participants' awareness, as well as positive perceptions about efficiency and applicability.*

Keywords*: Systemic impact analysis, Agile Requirements Engineering, Critical Requirements Engineering, Sustainability awareness*

Resumo. *Atualmente a Engenharia de Requisitos (ER) favorece a definição de requisitos claros e completos, possibilitando a entrega de funcionalidades adequadas e realmente necessárias. Entretanto, há um conjunto de aspectos socioeconômicos e valores que raramente são considerados. Devido à falta da análise sistêmica de impacto, softwares foram introduzidos na sociedade causando adversidades, desde inviabilização de empregos até segregação racial. Por essa razão, propomos um framework ágil de análise de impactos como parte da ER. O objetivo da abordagem é possibilitar a visualização dos impactos sistêmicos que o software pode ter, para que os engenheiros tenham a oportunidade de alterar o cenário previsto através de ajustes nos requisitos. Abordagens sistêmicas propostas anteriormente foram avaliadas e integradas na construção do framework. A proposta foi avaliada por estudantes e os resultados mostram efetividade na conscientização dos participantes, e ainda percepções positivas sobre eficiência e aplicabilidade.*

Palavras-chave*: Análise sistêmica de impacto, Engenharia de Requisitos ágil, Engenharia de Requisitos crítica, Consciência de sustentabilidade*

1. Introdução

As sociedades modernas são cada dia mais dependentes de sistemas de software, que estão na base de quase todos os aspectos da vida cotidiana [Deek et al. 2005]. Softwares influenciam o ambiente natural e socioeconômico através de efeitos de longo alcance sobre a forma como nos relacionamos, consumimos serviços e executamos tarefas. O processo de engenharia raramente explicita tais efeitos [Becker et al. 2016]. Como pilar de vários sistemas sociotécnicos, o software também é um fator-chave em sua sustentabilidade – a capacidade de suportar as atuais configurações econômicas, ambientais, sociais, técnicas e individuais [Duboc et al. 2019]. Todas as linhas de código têm não apenas implicações financeiras e técnicas, mas também consequências morais e éticas, uma vez que serviços de software moldam e participam do comportamento humano [Booch 2015] e é desejável que a sustentabilidade seja minimamente abalada pelos impactos do software.

O conceito de sustentabilidade é associado principalmente à ecologia e ao relacionamento entre os humanos e o planeta [Woodruff and Mankoff 2009]. Recentemente, a sustentabilidade de software emergiu como área de pesquisa no domínio da Engenharia de Software (ES) [Venters et al. 2014]. O impacto de um software em seu ambiente geralmente é determinado pela maneira como os engenheiros de software entendem seus requisitos [Becker et al. 2016]. Faz ou deveria fazer parte da Engenharia de Requisitos (ER) analisar e otimizar o impacto antes de introduzi-lo na sociedade. A ER deve articular e sintetizar da melhor forma possível as necessidades do mundo real para um projeto de software (de fora para dentro) e entender o impacto completo desse software ao retornar as necessidades atendidas à sociedade (de dentro para fora). Posto de forma simples, o relacionamento da ER com a sociedade é bidirecional [Ruhe et al. 2017].

Os engenheiros de software devem atuar de forma responsável frente as soluções tecnológicas que introduzem na sociedade e, por isso, se basear na perspectiva holística do pensamento sistêmico ao invés do *mindset* computacional de "resolver um problema para um cliente" [Easterbrook 2014], responsável por efeitos a longo prazo inesperados, como o processo de gentrificação (e consequentemente, disparidades raciais) que a plataforma Airbnb gerou em Nova York [Wachsmuth and Weisler 2018].

Nesse contexto, o problema tratado neste estudo é a falta de uma solução ágil para a análise de impactos socioeconômicos, que possa ser incorporada pelas empresas de desenvolvimento de software. Para propor a solução, o estudo se baseia em atividades e artefatos das abordagens sistêmicas *Critical Systems Thinking* e a metodologia DSRP, do inglês "Distinções, Estruturas, Relações e Perspectivas". A resolução deste problema é relevante pois fornece às empresas uma forma de aplicar o pensamento sistêmico crítico e decidir sobre pontos relacionados ao impacto dos sistemas que elas desenvolvem. Utilizando uma abordagem ágil que possibilite a visualização dos impactos sistêmicos que o software pode ter, a empresa adquire consciência e tem a oportunidade de alterar o cenário previsto através de ajustes nos requisitos.

O objetivo geral deste estudo é propor um framework ágil para análise sistêmica de impacto como parte da ER. Os objetivos específicos são: 1) elencar atividades e artefatos de abordagens sistêmicas e relacionadas a requisitos; 2) combinar atividades e artefatos selecionadas na definição de um novo framework; 3) promover uma avaliação do framework em relação ao objetivo geral.

O framework foi construído a partir de uma análise avaliativa de abordagens propostas anteriormente, em seguida, disponibilizado em um website, e finalmente avaliado por estudantes de Engenharia de Software (ES). A contribuição do trabalho é o framework proposto, como um meio para realizar a análise sistêmica de impactos do software no contexto do desenvolvimento ágil, e como um avanço na conscientização dos profissionais de ES em relação à responsabilidade que têm nos impactos do software sobre a sustentabilidade do sistema.

Este trabalho se estrutura da seguinte forma: na seção 2 são introduzidos conceitos relevantes ao estudo; na seção 3 trabalhos relacionados são apresentados; na seção 4 são discutidos os materiais e métodos do estudo; na seção 5 é feita a análise de integração de abordagens; na seção 6 a especificação de atividades do framework são descritas; na seção 7 os resultados são apresentados e na seção 8 tais resultados são discutidos. Por fim, a seção 9 traz as conclusões do estudo e trabalhos futuros propostos.

2. Fundamentação teórica

Nesta seção são apresentados conceitos relativos a ER e Métodos Ágeis. O Pensamento Sistêmico, Pensamento Sistêmico Crítico e DSRP são detalhados. Finalmente, a Sustentabilidade é apresentada como meta a ser alcançada através da análise de impactos.

2.1. Engenharia de requisitos (ER)

A Engenharia de Requisitos é a disciplina da ES que lida com o processo de elicitação e especificação dos requisitos de usuários. Os requisitos são a fundação de todo produto de software [Sommerville and Sawyer 1997], por proverem uma descrição precisa de necessidades e fornecerem uma forma de verificar a corretude da implementação [Easterbrook 1993]. Omissão, ambiguidade e erros na definição de requisitos levam a projetos que não satisfazem as necessidades dos usuários.

A detecção tardia de erros na especificação geram altos custos de retrabalho. Por isso há grande importância em validar os requisitos. Sommerville afirma que as etapas de validação (estudos de viabilidade, prototipação e revisão) se preocupam em encontrar problemas nos requisitos. Dessa forma, tais etapas se apresentam como ideais para a inserção da análise sistêmica de potenciais impactos dos requisitos.

2.2. Métodos ágeis

Alguns dos processos mais inovadores para desenvolvimento de software foram introduzidos pelo manifesto ágil [Beck et al. 2001], focados na eficiência, com entregas constantes e flexibilidade no escopo. Hoje, métodos ágeis como o Scrum, o Kanban e o Extreme Programming (XP), são considerados padrão na ES, dado seu potencial para operar em ambientes dinâmicos e competitivos [Stavru 2014].

2.3. Sustentabilidade dimensional

A sustentabilidade é inerentemente um conceito sistêmico que precisa ser entendido como um conjunto de dimensões [Kocak et al. 2015], definidas por:

- **Individual**: envolve a liberdade individual, a dignidade humana e a realização. Inclui a capacidade dos indivíduos de prosperar, exercer seus direitos e se desenvolver livremente.

- **Social**: abrange as relações entre indivíduos e grupos; as estruturas de confiança e comunicação mútuas em um sistema social e o equilíbrio entre interesses.
- **Econômica**: abrange aspectos financeiros e valor comercial. Envolve crescimento e liquidez de capital, questões de investimento e operações financeiras.
- **Técnica**: abrange a capacidade de manter e desenvolver sistemas artificiais (como software) ao longo do tempo. Refere-se à manutenção e evolução, resiliência e facilidade de transição do sistema.
- **Ambiental**: abrange o uso e administração dos recursos naturais, desde a produção de resíduos e consumo de energia até o equilíbrio dos ecossistemas e as preocupações com as mudanças climáticas [Penzenstadler et al. 2018].

2.4. Systems Thinking (ST)

Systems Thinking (traduzido como "pensamento sistêmico") é um conjunto de habilidades analíticas sinérgicas utilizadas para melhorar a capacidade de identificar e entender os sistemas, prever seu comportamento e propor modificações a eles para produzir os efeitos desejados [Arnold and Wade 2015].

Easterbrook argumenta que o ST fornece a ponte necessária do pensamento computacional para a prática da sustentabilidade, já que fornece ontologia de domínio para ponderar sobre sustentabilidade, uma base conceitual para refletir sobre mudança transformacional, e um conjunto de métodos para pensar criticamente sobre os impactos sociais e ambientais da tecnologia [Easterbrook 2014].

2.5. Critical Systems Thinking (CST)

O *Critical Systems Thinking* (ou "pensamento sistêmico crítico") reconhece que a teoria social e o pensamento sistêmico possuem forças e fraquezas complementares [Jackson 2001]. Ao cunhar o termo, o autor [Ulrich 2003] relata que encontrou um elo metodológico crucial entre o ST e a crítica no uso crítico das fronteiras do sistema. Para ele, o ST sem crítica é cego em relação aos seus julgamentos de fronteira subjacentes e suas implicações normativas; a crítica sem o pensamento sistêmico é ilimitada e, em última análise, vazia, pois seu objeto e contexto de aplicação permanecem arbitrários.

Segundo Elsawah et al., os métodos de CST são projetados para ajudar os praticantes a verem através de suas premissas e contemplarem os efeitos de seus julgamentos. Ao usar CST, o analista enquadra cada escolha feita como um julgamento de fronteira (determinação do que é incluído e excluído do processo de elicitação) [Elsawah et al. 2015]. Em seguida, o analista é engajado em uma prática reflexiva para pensar nas implicações da fronteira, identificar os vieses de seletividade envolvidos e questionar se há necessidade de rever os julgamentos.

2.6. DSRP

O DSRP é uma teoria proposta por Derek Cabrera [Cabrera and Cabrera 2018], e faz parte da quarta onda de *Systems Thinking* [Flemm 2019], o que a torna o que há de mais novo no campo. A teoria é uma simplificação dos conceitos tradicionais de ST, e propõe que os sistemas são constituídos por 4 padrões:

- **Distinção**: define que as "coisas" em um sistema se distinguem das outras ao redor. Distinção também se relaciona com a delimitação de fronteiras (o que está dentro e o que está fora do sistema modelado).

- **Estrutura**: define a existência de parte e todo. Todas as "coisas" em um sistema podem se dividir em partes ou serem agregadas a um todo.
- **Relação**: define a existência de ação e reação. As "coisas" se relacionam quando a ação de um lado provoca reação no outro.
- **Perspectiva**: define a existência de ponto e vista - quem vê e o que vê. Todas as "coisas" em um sistema podem ser o ponto ou a vista de uma perspectiva. Esse conceito tem um papel central no ST em geral, pois a mesma coisa pode ser vista de várias formas por diferentes atores.

3. Trabalhos relacionados

Os trabalhos relacionados discutidos nesta seção envolvem o contexto de inserção da solução e a incorporação de abordagens críticas ao processo de desenvolvimento de software como tentativa de alcançar a sustentabilidade, entender e otimizar os potenciais impactos sobre o sistema socioeconômico ao qual o software se integra.

Fraga e Barbosa constataram que a ausência de uma definição formal das atividades e produtos relacionados à ER em projetos que adotam métodos ágeis faz com que as organizações definam estas atividades de diversas maneiras [Fraga and Barbosa 2018]. Eles detectaram que os respondentes que consideram um formato de documentação pouco ou nada útil, em geral, raramente ou nunca o utilizam. Isso está relacionado a liberdade dos profissionais ágeis na escolha de processos, o que é considerado um ponto positivo para a proposta uma nova abordagem. Esse estudo é relevante por que descreve quais são as práticas já consolidadas no contexto de inserção da solução.

Easterbrook propõe que o ST é ideal para conectar o pensamento computacional à prática da sustentabilidade [Easterbrook 2014]. O autor declara que a sustentabilidade requer uma abordagem holística (que capture todo o sistema) e aponta pontos-chave que tal abordagem deve visar. As propostas de aplicação do pensamento sistêmico apresentadas nesse estudo são avaliadas na análise de integração do estudo atual.

Ferrario et al. partem do problema de que os valores incorporados ao software geralmente são invisíveis e subestimados, exceto quando as consequências catastróficas de sua violação se manifestam [Ferrario et al. 2016]. Dessa forma, almejam dar mais visibilidade à inter relação entre os valores pessoais e as escolhas feitas na ES. Eles incorporam os princípios da pesquisa de valores no processo de tomada de decisão, através do *Values-First Software Engineering* (VFSE), abordagem de ES que prioriza os valores humanos e refletem sobre as implicações de sua aplicação no desenvolvimento de software. A VFSE se apresenta como mais uma abordagem relacionada a identificação de perspectivas pessoais no desenvolvimento e portanto se faz adequada para análise no estudo atual.

Duboc et al. (2020) introduzem o conceito de engenharia de requisitos crítica ao desenvolverem um projeto para aplicar as Heurísticas de Sistemas Críticos (do inglês *Critical Systems Heuristics* - CSH, framework de CST) à ER com o objetivo de conscientizar criticamente sobre valores humanos, poder e política e questionar a seletividade das decisões tomadas [Duboc et al. 2020]. Os pesquisadores contemplaram a reflexão crítica que os frameworks de ER não proporcionam sozinhos. O uso das CSH na ER comprova que tal framework é favorável para abordar e orientar preocupações éticas dos softwares na sociedade, o que torna-o apropriado para ser avaliado no estudo atual como método para levantamento de pontos relevantes.

Duboc et al. (2019) defendem que os softwares devem ser projetados de forma a manter a sustentabilidade do sistema que integram [Duboc et al. 2019]. Para tal, a ER é a chave: ela torna possível a conscientização acerca da relação entre software e sustentabilidade. Os autores propõem um framework baseado em perguntas para a Conscientização de Sustentabilidade (*Sustainability Awareness* em inglês - SusA). Para verificar sua eficácia, uma avaliação foi feita através de um experimento de 2 semanas que envolveu 47 estudantes divididos em grupos. Os resultados mostraram que as perguntas proporcionaram discussões relevantes acerca da sustentabilidade do software, levando a *insights* e descobertas; além disso, auxiliaram na identificação de efeitos e cadeias de efeitos, levantando a importância de pensar nos cenários extremos através das 5 dimensões. O objetivo do SusA é fornecer um primeiro passo em direção à conscientização dos profissionais de software sobre os potenciais efeitos dos softwares na sustentabilidade.

4. Materiais e métodos

A pesquisa proposta neste artigo tem caráter qualitativo e aplicado. Aplicado por que tem por objetivo gerar conhecimentos para aplicação prática, dirigidos à solução de um problema; qualitativo pois a integração das abordagens busca detectar características não objetivas, como eficiência, compreensibilidade, aplicabilidade e reprodutibilidade. O restante desta seção apresenta as etapas necessárias para a realização deste estudo.

O trabalho tem o objetivo de avaliar e integrar abordagens sistêmicas e requisitos de software para então propor um novo framework que contemple a aplicabilidade da análise sistêmica de impacto no contexto ágil.

4.1. Análise de integração

A primeira parte do estudo analisou qualitativamente e discutiu as possíveis integrações das abordagens selecionadas. Isso significou selecionar as atividades mais alinhadas ao alcance da reflexão crítica e avaliar quais dessas atividades poderiam se complementar. A análise de integração foi detalhada na seção 5.

4.2. Especificação e construção do website

Após a análise de integração das abordagens obteve-se como produto o novo framework, formalmente definido através da descrição de todas as atividades e artefatos. Esta especificação foi detalhada na seção 6. Inspirado no manifesto Karlskrona (`https://www.sustainabilitydesign.org/`) o framework foi disponibilizado como um site na web.

O website `https://olharparaosistema.github.io/ops/` é o produto do trabalho que representa toda a proposta e contempla as instruções e artefatos necessários para que qualquer equipe de desenvolvimento seja capaz de entender e executar o framework. Para cada atividade proposta foi criado e apresentado também um modelo de exemplo, referente ao Airbnb. Cada item incluído no modelo foi explicado de forma bem detalhada, para favorecer a visão de "Como fazer".

4.3. Avaliação da proposta

Questionários disponibilizados no Google Forms e respondidos por estudantes de ES após demonstração ou aplicação prática do framework foram utilizados para verificar se a proposta cumpriu seus objetivos. A análise das respostas tornou possível avaliar se a solução

possui aplicabilidade no contexto ágil e quais foram as percepções dos estudantes. A avaliação foi detalhada na seção 7.

5. Análise de integração

O ponto de partida para propor as atividades do novo framework foi replicar o Sustainability Awareness (SusA) de Duboc et al. (2019) com algumas adaptações, levantadas por meio de pesquisas exploratórias sobre o estado da arte de ST, bem como análise de como o ST e o software foram relacionados nos estudos com objetivos correlatos. Na proposta final outras abordagens além do SusA foram integradas, como descreve-se abaixo:

5.1. Systems thinking "visual"

A construção do modelo do sistema foi considerada essencial no framework pois o ST é uma linguagem visual. "Palavras e frases devem vir uma de cada vez em ordem lógica e linear, já os sistemas acontecem inteiros de uma vez. Dessa forma, figuras funcionam melhor que palavras para a linguagem de sistemas, como possibilitam ver todas as partes da figura ao mesmo tempo" [Meadows 2008]. Um glossário de conceitos sistêmicos extraídos da obra de Meadows foi utilizado na primeira aplicação e, na reformulação, substituído pelos conceitos simplificados do DSRP.

5.2. Critical Systems Heuristics (CSH)

Para iniciar a construção do modelo é necessário identificar as fronteiras do sistema, ou quais as partes/entidades que serão consideradas na modelagem. Em apoio a essa atividade, propomos utilizar as 12 perguntas das CSH como norteadoras.

5.3. DSRP

Os padrões DSRP são a base da modelagem sistêmica realizada, e foram usados na análise de relações, subdivisões, variáveis de interesse e perspectivas.

5.4. Questionários do SusA

Para favorecer a reflexão dos participantes e a expansão de perspectivas, inicialmente mantivemos o uso de formulários no lugar das entrevistas. Porém, na primeira aplicação, 71,4% dos participantes não utilizaram o artefato. Com isso, a atividade de responder aos formulários das 5 dimensões foi considerada dispendiosa e removida da proposta. Grande parte das reflexões que as perguntas se propõem a criar já são possibilitadas pelo uso das CSH.

5.5. Diagrama de efeitos do SusA

A visualização final da análise sistêmica realizada foi sintetizada em diagramas de causa e efeito na proposta do SusA. No lugar do diagrama radar, propomos a utilização da ferramenta online Loopy (`https://ncase.me/loopy/`).

6. Especificação

Essa seção apresenta as atividades e instruções que compõem a proposta e exemplifica sua aplicação com o caso do Airbnb. O framework foi nomeado Olhar Para o Sistema (OPS), referindo-se à atividade principal que é uma forma de olhar para o sistema (construído

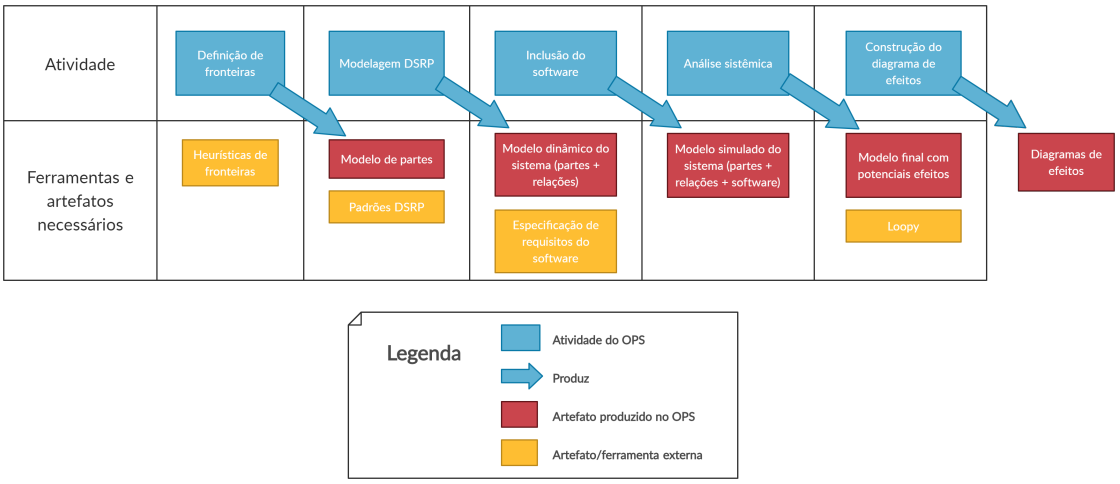

Figura 1. Visão geral da abordagem proposta

pelas percepções) para discutir e convergir sobre os efeitos. A sequência de passos para execução, bem como todos os artefatos necessários, foram representados na Figura 1, disponibilizados no website `https://olharparaosistema.github.io/ops/` junto com exemplos, e definidos como se segue:

6.1. Preparação

Para a execução do OPS é necessário reunir os analistas de requisitos responsáveis e, quando possível, stakeholders que dominem o contexto da solução. Isso por que os analistas de requisitos são responsáveis pela solução de software e compreendem o sistema em que ela será inserida; enquanto os stakeholders externos são fontes de informação mais próximas a tal sistema. Essas percepções serão o principal insumo para o modelo.

Para a execução correta do OPS é necessário um participante guia ou mediador. Além da exposição de informações e instruções também é preciso que o guia prepare os instrumentos (materiais de modelagem, apresentação dos requisitos).

6.2. Definição de fronteiras

A partir das percepções dos participantes, a primeira atividade busca delinear quais partes/entidades do sistema sociotécnico devem ser consideradas no processo.

As Heurísticas CSH se tratam de 12 perguntas que favorecem a identificação e reflexão sobre:

- **papeis**:beneficiários, tomadores de decisões, experts e testemunhas;
- **interesses**: propósito, recursos, expertises, oportunidades;
- **problemas**: medida de sucesso, ambiente de decisão, garantia de sucesso, espaço de reconciliação.

Com isso, mostram-se ideais para o objetivo da atividade. Durante a definição das fronteiras do sistema, o padrão Distinção já é contemplado. Ao fim da atividade Definição de fronteiras, tem-se um modelo estático de entidades que serão consideradas na análise (não inclui o software ainda), como mostra a Figura 2. Tudo que está fora desse escopo é considerado irrelevante para a análise.

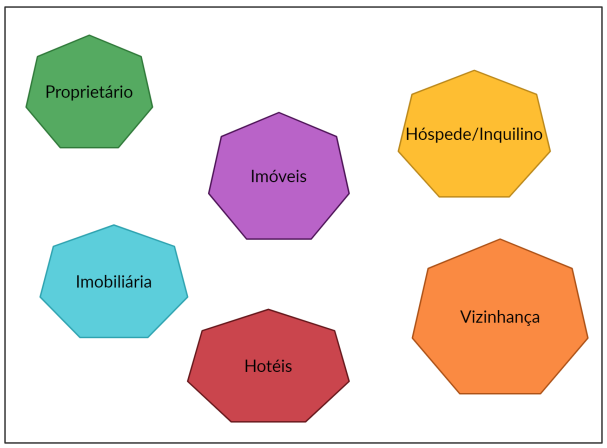

Figura 2. Airbnb: definição de fronteiras com CSH, com o mapeamento das principais entidades, incluindo beneficiários e partes afetadas negativamente.

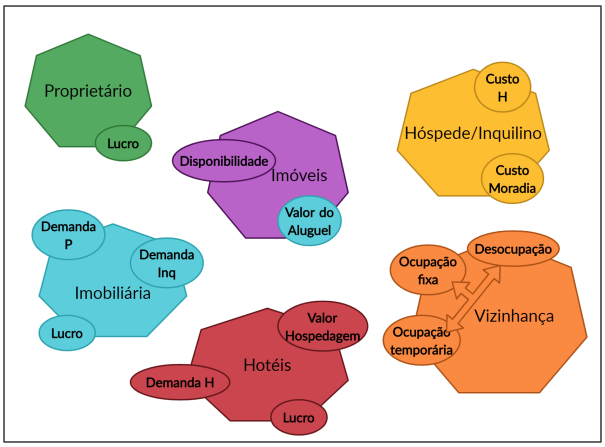

Figura 3. Airbnb: identificação de interesses proposta pelos padrões Estrutura e Perspectiva, e variáveis com comportamento temporal.

6.3. Modelagem DSRP

A ideia da modelagem DSRP é aplicar os 4 padrões ao modelo do sistema que foi iniciado. Como a distinção (desenho das fronteiras do sistema e entre as partes) foi aplicada pelas CSH, o próximo padrão a ser analisado é a Estrutura. É feita uma imersão nas partes definidas anteriormente para entender quais são seus interesses e subdivisões (se houver). É provável que as CSH já tenham introduzido na discussão ao menos uma das variáveis de interesse quando tratou de propósitos e medidas de sucesso.

A questão de entender os interesses é também uma aplicação do padrão Perspectiva. É preciso ver a perspectiva de cada entidade para entender quais são seus objetivos e/ou variáveis relevantes. Em muitos casos, com softwares comerciais, teremos entidades envolvendo demanda e receita como variáveis de interesse.

Para determinar as variáveis, é importante validar se o item escolhido possui comportamento temporal (é possível plotar em um gráfico, visualizar picos e quedas nessa variável?). Caso não possua, é provável que seja um interesse abstrato e difícil de quantificar, podendo ser irrelevante para a análise. A Figura 3 exemplifica a definição das variáveis de interesse em cada entidade dentro do modelo Airbnb.

Após modelar os interesses usando os padrões Estrutura e Perspectiva, é o momento de identificar as relações entre as partes. Por definição, as relações acontecem quando ações em uma variável geram reação em outra.

É importante observar que as relações são compostas por apenas 2 extremidades e tem ainda atributos que precisam ser discutidos e representados:

- **proporcionalidade**: aumento de um lado gera aumento do outro (relação direta) ou aumento gera queda (relação inversa). Tipos podem ser diferenciados por cor da linha ou linha pontilhada, por exemplo.
- **orientação**: deixa claro no modelo qual é a ação (origem) e a reação (destino), por exemplo, com setas.

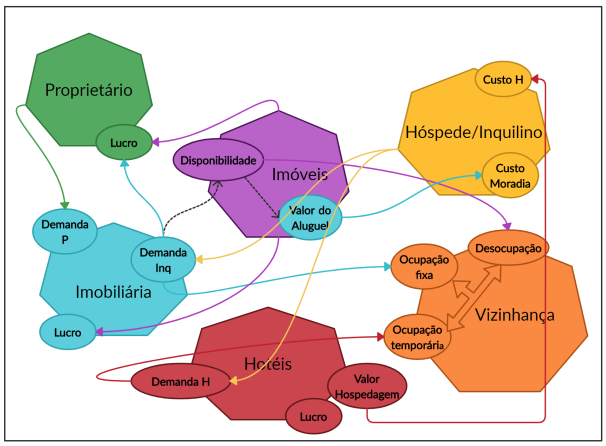

Figura 4. Airbnb: as relações entre as variáveis seguem o padrão Relação e determinam a parte "previsível" da dinâmica sistêmica.

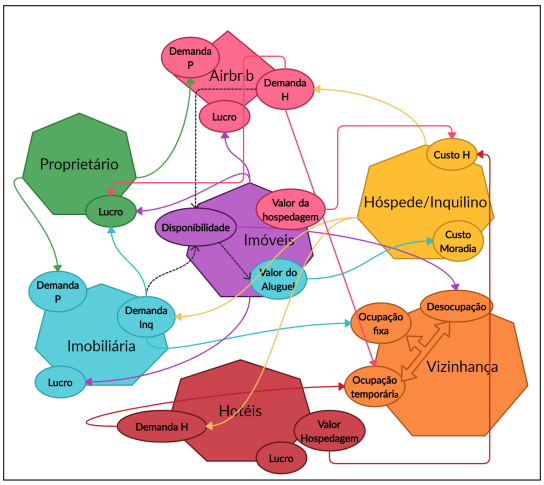

Figura 5. Airbnb: variáveis de interesse e relações imediatas do software se assemelham às já existentes em outras entidades.

- **demora**: tempo que a reação leva para acontecer após a ação na variável de origem. É considerado apenas durante o uso da ferramenta Loopy, onde é representado pela extensão da linha (linhas curtas são reações rápidas, linhas longas são reações demoradas).

A modelagem DSRP resulta no modelo dinâmico do sistema (Partes e Relações visíveis, com Perspectivas e Distinções coexistentes, apesar de invisíveis), como mostra o exemplo do Airbnb na Figura 4.

6.4. Inclusão do software

Para complementar o modelo do sistema é necessário trazer o software (definido pelos requisitos) e incluí-lo como entidade no sistema. Como todas as outras partes, também é preciso registrar as variáveis de interesse dentro do software e determinar as novas relações que sua inclusão no mundo real irá causar. Como estes efeitos da utilização do software são imediatos, não são complicados de identificar: em geral são o propósito do software sendo alcançado.

A etapa de inclusão do software resulta no modelo simulado do sistema (modelo dinâmico construído anteriormente + simulação do software como participante) conforme o exemplo na Figura 5.

6.5. Análise sistêmica

A análise sistêmica busca supor um cenário de uso contínuo do software dentro do sistema e detectar quais efeitos iriam surgir.

Algumas perguntas que podem guiar essa reflexão são:

- quais as demandas que seriam tomadas pelo software?
- como as variáveis do software que influenciam outras poderiam causar excesso ou escassez em qualquer ponto ou extremidade?

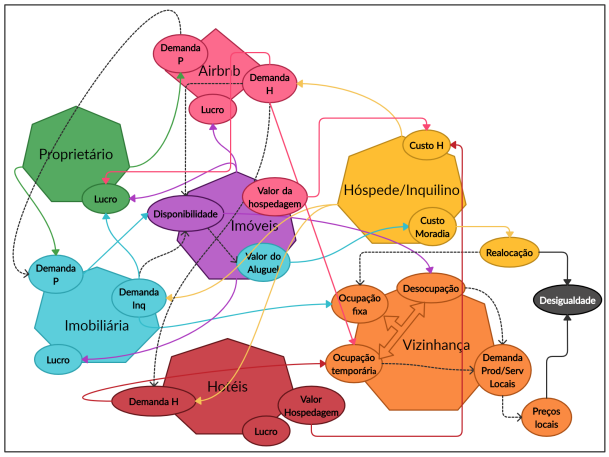

Figura 6. Airbnb: ao dividir a demanda com o mercado imobiliário e os hotéis (reduzindo a demanda dos mesmos) o Airbnb inicia a cadeia de efeitos sistêmicos.

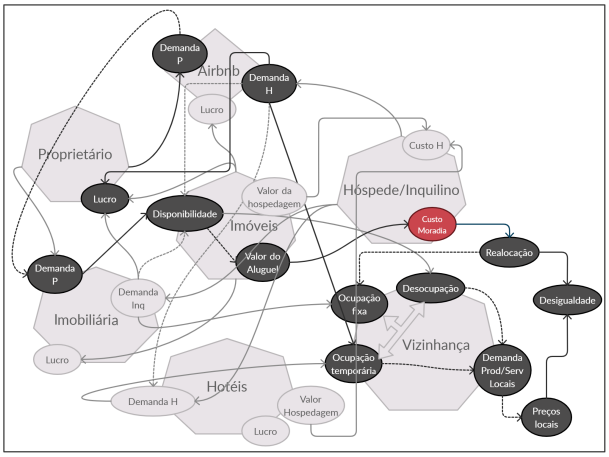

Figura 7. Airbnb: rastreando os efeitos identificados até as origens para destacar a cadeia de efeitos.

A Figura 6 exemplifica esse cenário de uso extremo no contexto do Airbnb. A análise sistêmica resulta no modelo com os potenciais efeitos que foram identificados.

6.6. Construção do diagrama de efeitos

Na etapa final, é feita uma síntese da análise sistêmica. É preciso rastrear os efeitos sistêmicos detectados na etapa anterior até suas origens (como ilustra a Figura 7) para modelar as cadeias de efeitos na ferramenta Loopy. A ferramenta foi escolhida por que possibilita a visualização da cadeia de efeitos em execução sequencial. É possível ainda dar "play" na cadeia de efeitos e exportar o diagrama por link ou arquivo. Além disso, a ferramenta possibilita a modelagem dos 3 aspectos mencionados: proporcionalidade, orientação e demora.

7. Avaliação e resultados

Para avaliar o framework, três turmas de alunos da graduação em ES na PUC Minas foram escolhidas. As turmas cursavam a disciplina Trabalho Interdisciplinar de Software (TIS) 4 e 5 durante o segundo semestre de 2020 e tinham um projeto de software em andamento. O domínio de atuação e experiência profissional e acadêmica dos alunos é descrita na Figura 8. Houve uma primeira aplicação em uma turma, e com os resultados, a proposta foi reformulada e, então, foi aplicada nas outras duas turmas.

7.1. Participantes e atividades na primeira aplicação (A1)

A proposta inicial contou com 4 artefatos base: "Slides de conscientização", usados para situar os participantes em uma etapa inicial de motivação; as "Heurísticas de fronteiras", para auxiliar na definição de fronteiras; "Glossário de conceitos sistêmicos" e "Formulários das 5 dimensões" do SusA, para incentivar a discussão/reflexão através das 5 dimensões e expandir as perspectivas. A A1 teve 14 participantes.

A A1 iniciou com uma apresentação breve (20 minutos) da proposta e do OPS, e os alunos foram instruídos a aplicar as atividades do framework no projeto de software em

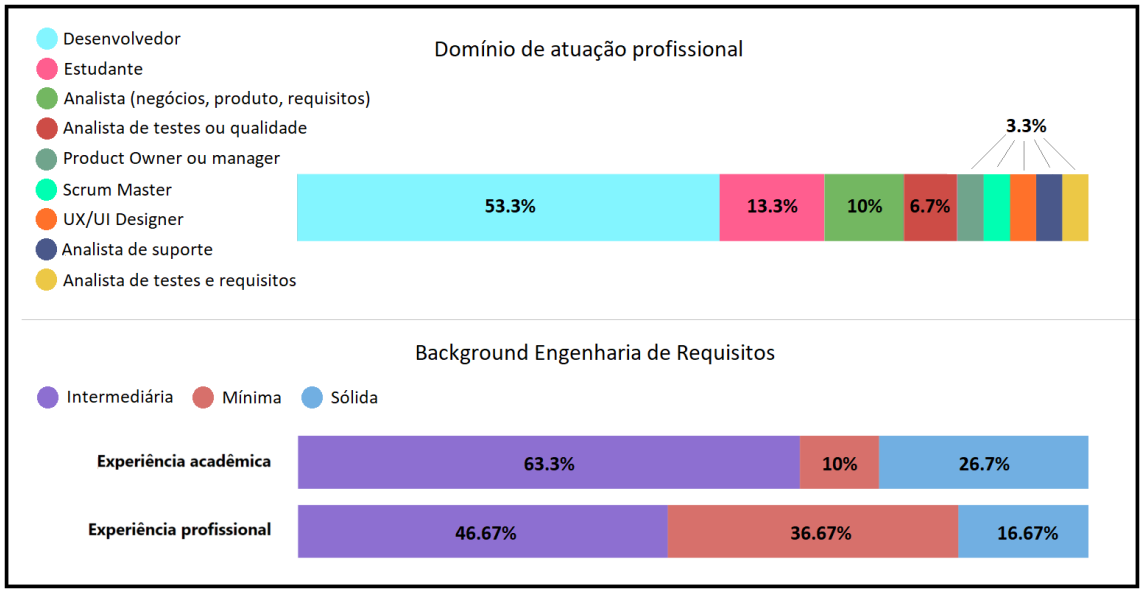

Figura 8. Caracterização do background dos participantes

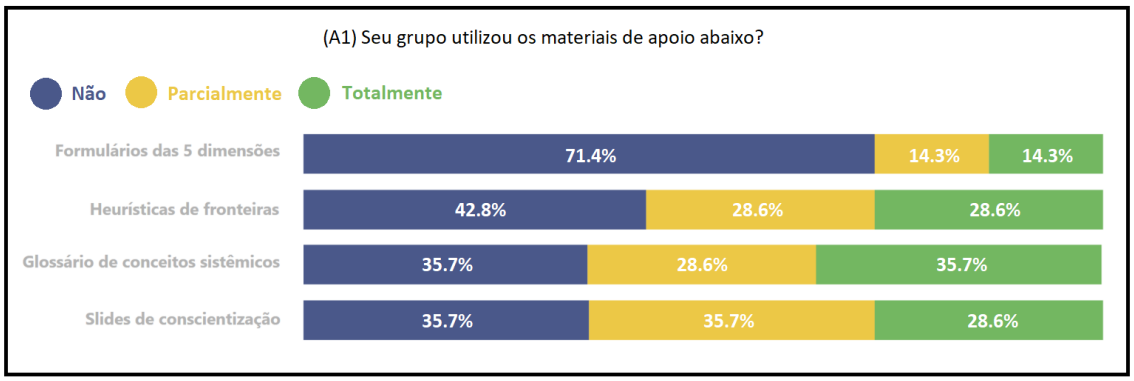

Figura 9. Utilização de artefatos na A1

andamento e entregar um diagrama final de efeitos. Os alunos se orientaram pelo website para fazer as atividades. Das 6 equipes respondentes, 3 chegaram ao diagrama final após o prazo de 2 semanas. Apenas uma delas afirmou ter usado todos os artefatos. 42,8% dos respondentes afirmaram ter usado nenhum ou somente um artefato. A baixa utilização de artefatos é ilustrada na Figura 9. Os formulários foram desconsiderados pela maioria de 71% dos participantes, enquanto os demais artefatos variaram entre 35 e 43%.

7.2. Participantes e atividades na segunda aplicação (A2)

Após a A1, alguns *insights* foram levantados sobre possíveis problemas e o framework foi reformulado. Dos 4 artefatos anteriormente mapeados, apenas as heurísticas de fronteiras foram mantidas. Além disso, houve a inclusão do DSRP e da ferramenta Loopy na proposta. A A2 teve 16 participantes.

A A2 consistiu em uma apresentação aprofundada (60 minutos) que contemplou uma demonstração detalhando todas as atividades propostas, desde a modelagem de fronteiras até o diagrama de causas e efeitos. Os participantes não executaram as ativi-

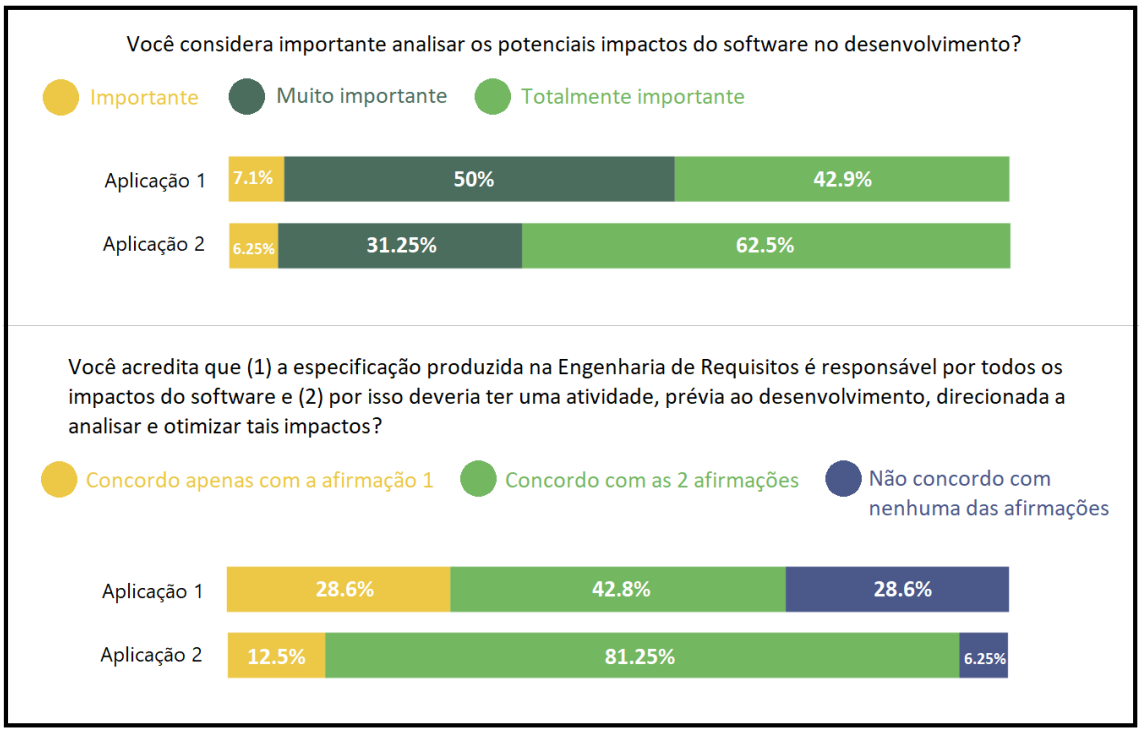

Figura 10. Resultados das perguntas referentes à conscientização

dades independentemente como na primeira aplicação, apenas observaram a aplicação, enquanto discutiam e tiravam eventuais dúvidas. Houve grande foco em instruções claras e na execução passo a passo, o que possibilitou compreensão e resultados melhores.

7.3. Resultados das aplicações

Os formulários enviados aos participantes contemplavam a avaliação nos aspectos: conscientização do participante, eficiência do framework, aplicabilidade do framework e percepções sobre o framework e a forma que ele foi introduzido/apresentado.

7.3.1. Conscientização

Duas questões de conscientização foram direcionadas aos participantes de ambas aplicações. A primeira questão versa sobre a percepção de importância pelo participante, enquanto a segunda avalia se o participante enxerga a necessidade da análise sistêmica, e entende o papel fundamental da ER na questão tratada. A Figura 10 ilustra os resultados.

A A2 teve um resultado excelente em conscientização, mostrando que a demonstração foi efetiva para abordar a questão de pensar nos impactos do software antes do desenvolvimento. Como não houve aplicação prática e toda a avaliação na A2 se trata de percepções, a conscientização é a percepção mais importante a se alcançar, uma vez que é o ponto de partida para engajar os estudantes na questão tratada.

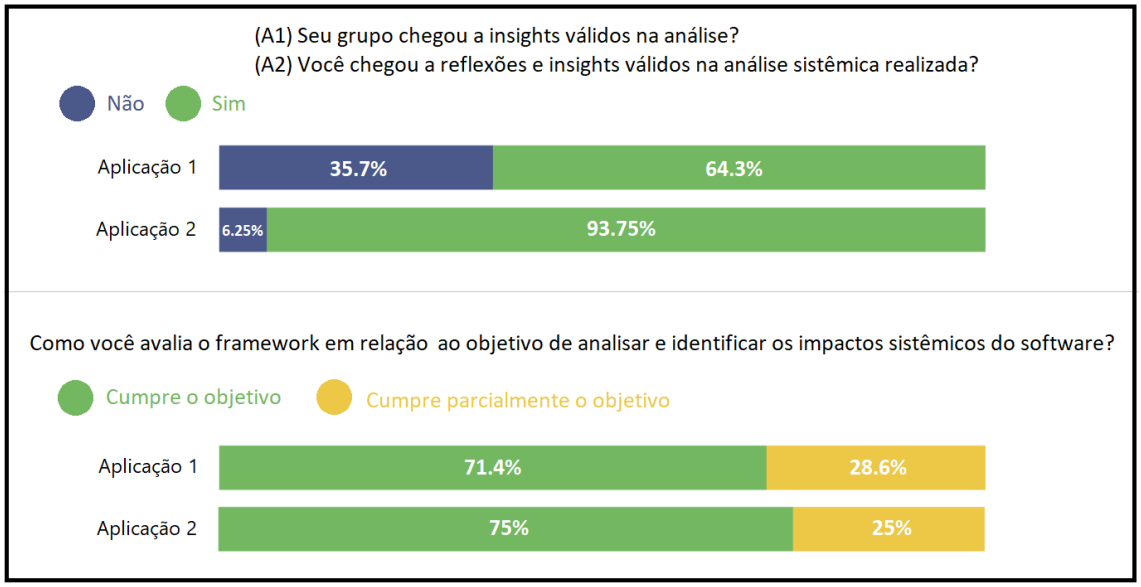

Figura 11. Resultados das perguntas referentes à eficiência

7.3.2. Eficiência

Duas questões foram feitas aos participantes com o objetivo de avaliar a eficiência da proposta, e seus resultados foram apresentados na Figura 11. A primeira avalia se a proposta foi capaz de gerar *insights* e reflexões nos participantes, que é o primeiro passo no caminho da análise sistêmica eficaz. A segunda busca avaliar como o participante percebe a conformidade da proposta com seu objetivo inicial.

Como a A1 contou com a aplicação prática, essas questões foram importantes para determinar a incerteza apresentada pelos participantes da A1 e a necessidade de reformulação. Dados os resultados razoáveis de conscientização obtidos na A1, era esperado que a eficiência também fosse ficar comprometida na avaliação, já que não é tão simples ver utilidade em algo cujo propósito não se entende por completo.

7.3.3. Aplicabilidade

A Figura 12 apresenta as respostas da pergunta que buscou avaliar a aplicabilidade do framework e determinar as percepções sobre a possibilidade de levar a proposta ao ambiente acadêmico ou profissional. Em ambos os casos mais de 90% dos participantes consideraram possível essa aplicação.

7.3.4. Percepções pessoais

Foi solicitado que os participantes avaliassem a "experiência" de conhecer a proposta. Essa pergunta é fundamental para determinar as formas mais eficientes de alcançar pessoas interessadas. A experiência na A1 foi o uso do website. Já na A2, o website não foi apresentado aos participantes (apenas disponibilizado o link nas referências), então a demonstração foi avaliada.

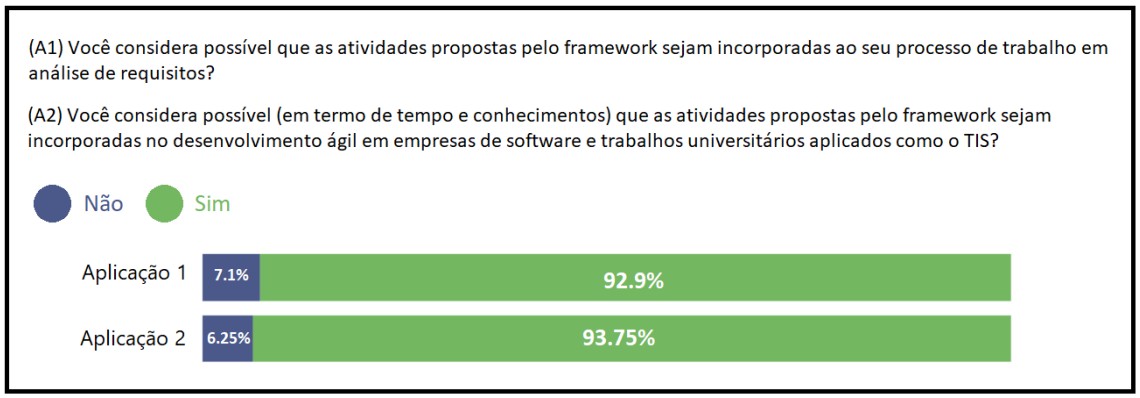

Figura 12. Resultados da pergunta referente à aplicabilidade

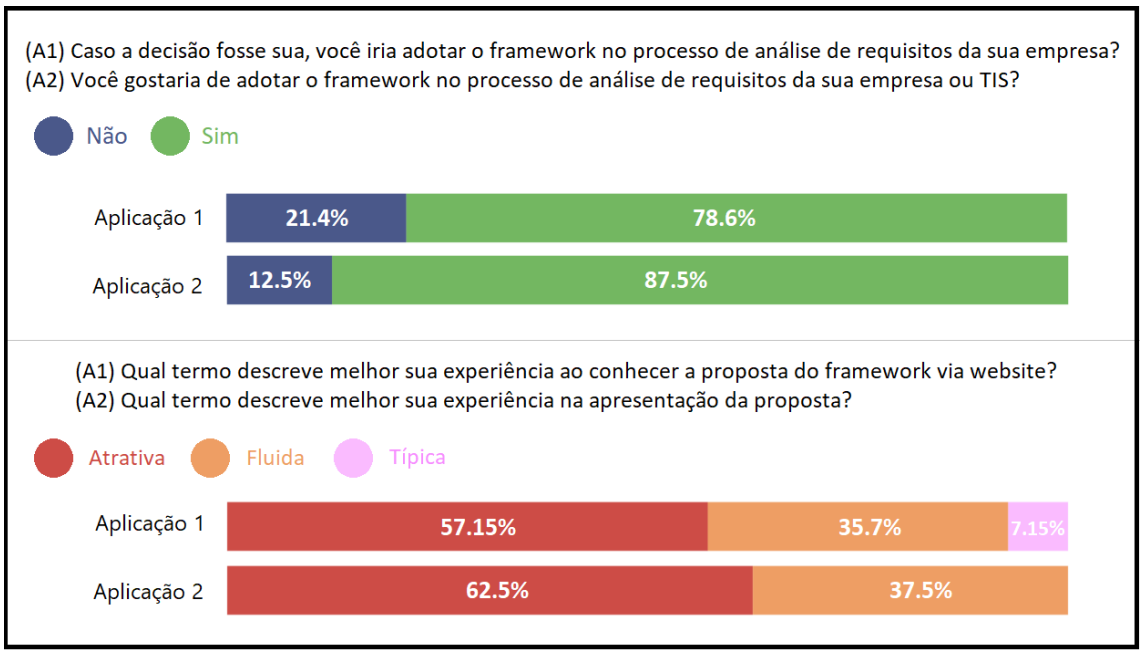

Figura 13. Resultados da pergunta referentes à percepções pessoais

Outra questão proposta busca avaliar se o participante gostaria de aplicar o framework em sua empresa ou no trabalho interdisciplinar. Os resultados de ambas as questões são apresentados na Figura 13. Ambas as perguntas tiveram pouca variação entre as duas aplicações e mostram percepções positivas para a maioria.

7.4. Dúvidas

Na A1, 42,8% dos participantes deixaram dúvidas sobre o framework. O principal ponto levantado foi a ausência de exemplos no website e falta de clareza sobre as instruções de execução das atividades. As 3 sugestões de melhorias recebidas (21,4% dos participantes registraram) tangiam a inclusão de exemplos no website (que foram inseridos antes da A2).

Na A2, apenas 2 participantes (12,5%) registraram dúvidas, sendo a primeira se já houve aplicação do framework em algum processo de desenvolvimento e, a segunda, como realizar a atividade de identificar as partes afetadas. Apenas 1 (6,25%) participante

registrou uma percepção adicional no campo aberto. Segundo ele, é difícil encontrar todas as partes que serão afetadas e por isso a aplicação se torna volátil e propensa a gerar dados inconsistentes.

8. Discussão dos resultados

As perguntas relacionadas à caracterização dos participantes ajudaram a assegurar que a experiência deles com requisitos era suficiente para avaliar o framework. Tanto no contexto acadêmico (63.3%) quanto no profissional (46.6%), o mais comum foi a experiência "intermediária".

Em ambas as aplicações, a experiência de conhecer o framework foi considerada "atrativa" por mais da metade dos participantes, seguida por "fluida". Apenas um participante considerou a experiência "típica". Outras opções disponíveis eram "inadequada" e "complicada". Este resultado é positivo pois mostra que a proposta teve impressão positiva e despertou interesse nos estudantes.

8.1. A1 - apresentação curta, execução guiada pelo website

As respostas relacionadas à A1 trouxeram resultados controversos em questão de conscientização: apesar de todos atribuírem algum grau de importância à tarefa de analisar os impactos sistêmicos do software no processo de desenvolvimento (sendo que 92,9% atribuíram "muito importante" ou "totalmente importante"), apenas 42,9% concordaram que deveria haver uma atividade prévia ao desenvolvimento para identificar e tratar tais impactos. Essa pergunta apontou também que 28,6% dos respondentes nem sequer concordam que a especificação produzida na ER é responsável por todos os impactos do software. Estes dados apontam que a conscientização na A1 não foi efetiva para grande parte dos participantes, e portanto, a necessidade de reformular.

Apesar disso, em questão de eficiência e aplicabilidade, 64,2% afirmaram ter chegado a *insights* válidos na análise feita, 93% consideraram possível incorporar as atividades no processo de ER e 78,6% afirmaram que adotaria o framework no processo de ER de sua empresa. Todos consideraram que o framework cumpre o objetivo (entre eles, 28,5% declararam que cumpre de forma parcial).

Como a A1 contou com a execução do framework por 3 grupos de alunos, foi possível avaliar o uso dos artefatos por eles. 42,8% afirmaram não ter usado nenhum ou somente um artefato. Isso indicou a necessidade de reavaliar a necessidade de cada artefato na proposta. Diagramas de efeitos foram enviados por 3 dos 6 grupos que responderam ao questionário. Dois dos diagramas contaram com identificação de conceitos apresentados, motivações, relacionamentos entre as partes e a identificação de cadeias de efeitos imediatos. Mas não houve identificação de efeitos sistêmicos.

Os resultados da A1 sugerem confiança dos participantes na proposta, mas pouca eficiência na aplicação orientada pelo website. As dúvidas e percepções (registradas por 42,8%) apontaram a necessidade de tornar as instruções mais claras e fornecer exemplos na orientação. Tais melhorias foram incorporadas à proposta antes da A2.

8.2. A2 - demonstração de execução

A proposta reformulada com a demonstração detalhada trouxeram resultados bastante positivos e mais conclusivos em relação à conscientização. Novamente todos atribuíram

algum grau de importância para a análise dos impactos sistêmicos do software no desenvolvimento. 81,25% dos participantes concordaram que deve haver uma atividade direcionada a analisar e otimizar os impactos antes do desenvolvimento e que a ER é responsável por todos os impactos do software (quase o dobro em relação à A1).

Em relação à eficiência e aplicabilidade, 93,7% afirmaram ter chegado a *insights* válidos durante a execução, um resultado também expressivo em relação à A1, que obteve 64,2%. Novamente todos afirmaram que o framework cumpre o objetivo (de forma parcial para 25%). A maioria de 93,7% considera possível incorporar as atividades propostas em empresas de desenvolvimento/trabalhos acadêmicos e a também maioria de 87,5% afirmou que gostaria de adotar o framework em sua empresa ou no trabalho interdisciplinar.

Dúvidas foram registradas por apenas 12,5% dos participantes, um número consideravelmente menor que o da A1. Uma vez que o exemplo foi apresentado, a preocupação relatada foi a identificação de partes afetadas, que está atribuída a atividade de aplicação das CSH. Isso indica a necessidade de aprofundar mais nessa atividade durante a exposição e relacionar de forma mais clara as perguntas com as fronteiras definidas.

De forma geral, os resultados de ambas as aplicações mostram que as pessoas reconhecem que o framework tem um objetivo relevante e aplicabilidade. Mas a A1 teve a falha do excesso de artefatos e ausência de exemplos, assim, possibilitou que os alunos modelassem o sistema, mas sem chegar a efeitos sistêmicos. A A2 corrigiu essas falhas, porém não contou com execução prática dos participantes.

9. Conclusões e trabalhos futuros

Este trabalho se relaciona, de um lado, a uma prática relativamente nova e até então totalmente acadêmica que é o de análise sistêmica para alcançar a sustentabilidade em software. De outro lado, se relaciona ao universo do desenvolvimento ágil, que é bastante presente nas empresas desenvolvedoras de software. De acordo com o levantamento inicial feito neste estudo, a intercessão entre esses dois lados é inexplorada.

A contribuição deste estudo é uma primeira tentativa de definir de forma explícita uma abordagem ágil, especificamente para fazer análise sistêmica dos requisitos de software. O website é um produto do trabalho que possibilita o acesso de qualquer pessoa interessada em realizar análise sistêmica de requisitos de software. Há uma seção no website para coleta de feedbacks, que torna possível fazer uma melhoria contínua da proposta com base em feedbacks recebidos / experiências de utilização.

Em relação às definições iniciais do estudo, ele foi capaz de cumprir seus objetivos determinados e entregar o resultado esperado.

A avaliação do framework foi dividida em dois formatos: a primeira aplicação contou com a execução guiada pelo website, e foi essencial por que apontou a necessidade de reformulação da proposta em três aspectos principais: pouca clareza de instruções, ausência de exemplo, excesso de artefatos. A segunda aplicação contou com a demonstração de execução já partindo da proposta reformulada e obteve resultados melhores, que indicam que esse formato é mais adequado para levar a proposta a pessoas que ainda não foram conscientizadas.

A conscientização sobre o papel da ER nos impactos do software foi realizada nas aplicações, mas é coberta de forma superficial no website (apenas nas seções introdutórias, e não dentro das 5 atividades propostas). Espera-se que as pessoas que buscam os meios para fazer análise sistêmica nos requisitos de software já tenham as motivações e objetivos estabelecidos.

A conscientização foi um aspecto levado em consideração na avaliação por que era necessário que os participantes compreendessem a motivação por trás da proposta. Além disso, como são engenheiros de software em formação, é válido que sejam conscientes em relação à responsabilidade pelos impactos do software e possam atuar. Por esse motivo, os resultados de conscientização são de extrema relevância para o estudo.

Para trabalhos futuros, seria ideal levar o OPS a um grupo de empresas no formato de workshop (que favorece tanto a conscientização quanto a aplicação prática) para que seja possível avaliar:

- o grau de conscientização de gerentes, analistas e desenvolvedores, que possuem um olhar mais crítico com relação a gasto de recursos e mudanças de requisitos.
- a efetividade de execuções independentes, guiadas pela demonstração inicial no workshop e pelo website.

Com o desenvolvimento de tal trabalho seria possível avançar para melhorar a proposta como um todo e assegurar a aplicabilidade ágil, rumo a uma proposta mais aplicável e com horizontes expandidos.

Referências

Arnold, R. D. and Wade, J. P. (2015). A definition of systems thinking: A systems approach. *Procedia Computer Science*, 44:669–678.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., and Thomas, D. (2001). Agile manifesto. Disponível em `http://agilemanifesto.org/`. Acesso em 10/04/2020.

Becker, C., Betz, S., Chitchyan, R., Duboc, L., Easterbrook, S. M., Penzenstadler, B., Seyff, N., and Venters, C. C. (2016). Requirements: The key to sustainability. *IEEE Software*, 33(1):56–65.

Booch, G. (2015). The future of software engineering (seip keynote). *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, 2:3–3.

Cabrera, D. and Cabrera, L. (2018). *Systems Thinking Made Simple: New Hope for Solving Wicked Problems*. Plectica Publishing.

Deek, F. P., McHugh, J. A. M., and Eljabiri, O. M. (2005). *Strategic Software Engineering: An Interdisciplinary Approach*. Auerbach Publications.

Duboc, L., Betz, S., Penzenstadler, B., Akinli Kocak, S., Chitchyan, R., Leifler, O., Porras, J., Seyff, N., and Venters, C. C. (2019). Do we really know what we are building? raising awareness of potential sustainability effects of software systems in requirements engineering. *2019 IEEE 27th International Requirements Engineering Conference (RE)*, pages 6–16.

Duboc, L., McCord, C., Becker, C., and Ahmed, S. I. (2020). Critical requirements engineering in practice. *IEEE Software*, 37(1):17–24.

Easterbrook, S. (1993). Negotiation and the role of the requirements specification. *Cognitive Science Research Reports*, 197.

Easterbrook, S. (2014). From computational thinking to systems thinking: A conceptual toolkit for sustainability computing. *ICT for Sustainability 2014, ICT4S 2014*, pages 235–244.

Elsawah, S., Ryan, M., and Mclucas, A. (2015). Beyond why to what and how: the use of systems thinking to support problem formulation in systems engineering applications. *The 21st International Congress on Modelling and Simulation (MODSIM2015)*.

Ferrario, M. A., Simm, W., Forshaw, S., Gradinar, A., Smith, M. T., and Smith, I. (2016). Values-first se: Research principles in practice. *IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C)*, pages 553–562.

Flemm, R. (2019). Scrum.org: Systems thinking, episode #0 what is systems thinking and why should i care? Disponível em `https://www.scrum.org/resources/blog/systems-thinking-episode-0-what-systems-thinking-and-why-should-i-care`. Acesso em 17/08/2020.

Fraga, B. S. and Barbosa, M. W. (2018). Uma investigação sobre como os profissionais brasileiros realizam atividades de engenharia de requisitos em projetos ágeis. *Revista Interdisciplinar Científica Aplicada*, 12(3):82–100.

Jackson, M. (2001). Critical systems thinking and practice. *European Journal of Operational Research*, 128:233–244.

Kocak, S., Becker, C., Betz, S., Calero, C., Chitchyan, R., Duboc, L., Easterbrook, S., Mahaux, M., Penzenstadler, B., Rodriguez-Navas, G., Salinesi, C., Seyff, N., and Venters, C. C. (2015). The karlskrona manifesto for sustainability design. Disponível em `https://www.sustainabilitydesign.org/karlskrona-manifesto/`. Acesso em 10/04/2020.

Meadows, D. H. (2008). *Thinking in Systems: A Primer*. Chelsea Green Publishing.

Penzenstadler, B., Duboc, L., Venters, C. C., Betz, S., Seyff, N., Wnuk, K., Chitchyan, R., Easterbrook, S. M., and Becker, C. (2018). Software engineering for sustainability: Find the leverage points! *IEEE Software*, 35(4):22–33.

Ruhe, G., Nayebi, M., and Ebert, C. (2017). The vision: Requirements engineering in society. *2017 IEEE 25th International Requirements Engineering Conference (RE)*, pages 478–479.

Sommerville, I. and Sawyer, P. (1997). *Requirements engineering: a good practice guide*. John Wiley and Sons.

Stavru, S. (2014). A critical examination of recent industrial surveys on agile method usage. *Journal of Systems and Software*, 94.

Ulrich, W. (2003). Beyond methodology choice: Critical systems thinking as critically systemic discourse. *Journal of The Operational Research Society*, 54:325–342.

Venters, C., Jay, C., Lau, L., Griffiths, M., Holmes, V., Ward, R., Austin, J., Dibsdale, C., and Xu, J. (2014). Software sustainability: The modern tower of babel. *CEUR Workshop Proceedings*, 1216:7–12.

Wachsmuth, D. and Weisler, A. (2018). Airbnb and the rent gap: Gentrification through the sharing economy. *Environment and Planning A: Economy and Space*, 50(6):1147–1170.

Woodruff, A. and Mankoff, J. (2009). Environmental sustainability. *IEEE Pervasive Computing*, 8(1):18–21.