UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

ANDRESSA HATSUE IWAZAKI DA SILVA

CONCEITUAÇÃO SOBRE NECESSIDADES DOS CLIENTES E PROPOSIÇÃO DE UMA TÉCNICA DE ASSOCIAÇÃO COM REQUISITOS DE SOFTWARE

CURITIBA

2024

ANDRESSA HATSUE IWAZAKI DA SILVA

CONCEITUAÇÃO SOBRE NECESSIDADES DOS CLIENTES E PROPOSIÇÃO DE UMA TÉCNICA DE ASSOCIAÇÃO COM REQUISITOS DE SOFTWARE

Conceptualization of Customer Needs and Proposal of a Technique for Associating Them with Software Requirements

Dissertação apresentada como requisito para obtenção do título de Mestre em Ciência – Área de Concentração: Engenharia de Computação do Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial (CPGEI) da Universidade Tecnológica Federal do Paraná (UTFPR).
Orientador: Prof. Dr. Paulo Cézar Stadzisz.

CURITIBA

2024

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Campus Curitiba

ANDRESSA HATSUE IWAZAKI DA SILVA

CONCEITUAÇÃO SOBRE NECESSIDADES DOS CLIENTES E PROPOSIÇÃO DE UMA TÉCNICA DE ASSOCIAÇÃO COM REQUISITOS DE SOFTWARE

Trabalho de pesquisa de mestrado apresentado como requisito para obtenção do título de Mestre Em Ciências da Universidade Tecnológica Federal do Paraná (UTFPR). Área de concentração: Engenharia De Computação.

Data de aprovação: 25 de Outubro de 2024

Dr. Paulo Cezar Stadzisz, Doutorado - Universidade Tecnológica Federal do Paraná
Dr**.** Andrey Ricardo Pimentel, Doutorado - Universidade Federal do Paraná (Ufpr)
Dr. Cesar Augusto Tacla, Doutorado - Universidade Tecnológica Federal do Paraná
Dr. Jean Marcelo Simao, Doutorado - Universidade Tecnológica Federal do Paraná

Documento gerado pelo Sistema Acadêmico da UTFPR a partir dos dados da Ata de Defesa em 09/12/2024.

Dedico este trabalho a Deus, à minha família e a todos que contribuíram para a sua realização.

AGRADECIMENTOS

Primeiramente, agradeço a Deus, que sempre cuida de mim e coloca em meu caminho pessoas que contribuem para minha evolução pessoal.

Meu sincero agradecimento a todos que acompanharam minha jornada no mestrado e que, direta ou indiretamente, contribuíram para a realização deste projeto.

Sou profundamente grata à minha família Iwazaki, especialmente à minha filha Angelina e o meu irmão Anderson, que sempre me encorajaram e apoiaram nas minhas decisões. A minha filha, com sua luz e sabedoria, me ensinou o verdadeiro significado de parceria, amor e sucesso, tornando-se uma fonte de inspiração constante em minha vida.

Finalmente, expresso minha mais sincera gratidão ao meu orientador, Prof. Dr. Paulo Cézar Stadzisz, que me ensinou uma das lições mais valiosas: “aprender a aprender” (Jean Piaget), com seu apoio contínuo, paciência e confiança. Também à Universidade Tecnológica Federal do Paraná (UTFPR), campus Curitiba, por tornar esta conquista possível.

“Toda ação humana, quer se torne positiva ou negativa, precisa depender de motivação”.
(DALAI LAMA).

RESUMO

Requisitos de Software é considerada uma importante atividade na Engenharia de Software, dado que a sua qualidade reflete no sucesso das atividades posteriores para o desenvolvimento de software e, principalmente, do produto no mercado. Esta atividade envolve a especificação técnica de requisitos funcionais e não funcionais do futuro produto e deveria responder à real demanda dos clientes para que se possa garantir que os requisitos estão corretos e completos. Por mais que exista um grande número de pesquisas e proposições na área, a atividade de Requisitos de Software continua a ser um grande desafio para as equipes de desenvolvimento de software, pois as abordagens existentes são insuficientes para um entendimento claro das demandas dos clientes e seu mapeamento para requisitos. O objetivo desta pesquisa é contribuir para o avanço da abordagem *Problem Based Software Requirements Specification* (PBSRS) em desenvolvimento na Universidade Tecnológica Federal do Paraná (UTFPR) por meio do detalhamento dos conceitos, da elaboração de uma ontologia e da proposição de uma técnica de especificação das “Necessidades dos Clientes” e de sua associação com os “Requisitos de Software”, visando tornar a abordagem mais precisa e prática aos profissionais. Por meio de um caso de estudo, gerou-se um diagnóstico que demonstrou a aplicabilidade dos conceitos e da técnica propostos.

Palavras-chave: Engenharia de Requisitos; Requisitos de Software; Necessidades dos Clientes.

ABSTRACT

Software Requirements is considered an important activity in Software Engineering, as its quality reflects the success of subsequent activities for software development and, most importantly, the product in the market. This activity involves the technical specification of functional and non-functional requirements of the future product and should respond to the real demand of customers to ensure that the requirements are correct and complete. Despite the large number of research and proposition in the area, the activity of Software Requirements continues to be a major challenge for software development teams, as existing approaches are insufficient for a clear understanding of customer demands and their mapping to requirements. The objective of this research is to contribute to the advancement of the Problem Based Software Requirements Specification (PBSRS) approach under development at the Federal University of Technology – Paraná (UTFPR), by means of the detailing of concepts, the development of an ontology, and the proposal of technique for specifying “Customer Needs” and associating them with “Software Requirements”, aiming to make the approach more precise and practical for professionals. Through a case study, a diagnosis was generated that demonstrated the applicability of the proposed concepts and techniques.

Keywords: Requirements Engineering; Software Requirements; Customer Needs.

LISTA DE ILUSTRAÇÕES

Figura 1 - Diagrama de ator do *Media Shop*.....29
Figura 2 - Diagrama de explicação do ator *Media Shop* 29
Figura 3 - Diagrama de ator do *Media Shop*.....30
Figura 4 - Diagrama de explicação para o *Media Shop*.....31
Figura 5 - *The Elements of Value* 33
Figura 6 - Ilustração dos domínios da abordagem AD.....36
Figura 7 - Ilustração dos domínios da abordagem PBSRS 37
Figura 8 - Organização da PBSRS 38
Figura 9 - Esquema da proposta de ontologia para Requisitos de Software.....50
Figura 10 – Esquema de conceituação de *Stakeholder*.....53
Figura 11 – Esquema de conceituação de Domínio dos Problemas 56
Figura 12 – Esquema de conceituação de Domínio das Necessidades 60
Figura 13 – Esquema de conceituação de Domínio dos Requisitos 64
Figura 14 – Ilustração dos domínios 66
Figura 15 – Visão geral da técnica proposta.....78
Figura 16 – *Glance* do *marketplace* 87

LISTA DE TABELAS

Tabela I - Representação dos itens do diagrama 28
Tabela II - Exemplo de tabela de incidência NE X RE 69
Tabela III - Exemplo de tabela de incidência PO X NE 70
Tabela IV - Tabela de incidência PO X NE.....82
Tabela V - Tabela Análise de Cobertura de Problemas.....90
Tabela VI - Tabela Análise de Cobertura das Necessidades 92

LISTA DE ABREVIATURAS E SIGLAS

CAs	*Customers Attributes*
DP	*Design Parameters*
FRs	*Funcional Requirements*
GORE	*Goal Oriented Requirements Engineering*
IEEE	*Institute of Electrical and Electronics Engineers*
MIT	*Massachusetts Institute of Technology*
OPM	*Object Process Methodology*
PBSRS	*Problem Based Software Requirements Specification*
PVs	*Process Variables*
RDF	*Resource Description Framework*
RH	Recursos Humanos
SWEBOK	*Software Engineering Body of Knowledge*
UTFPR	Universidade Tecnológica Federal do Paraná
W3C	*World Wide Web Consortium*
XP	*Extreme Programminng*

SUMÁRIO

1 INTRODUÇÃO 13
1.1 Contexto de Pesquisa 13
1.2 Problemas e Questões de Pesquisa 17
1.3 Motivações para a Pesquisa 19
1.4 Objetivo da Pesquisa 20
1.5 Método de Pesquisa Empregado 20
1.6 Estrutura do Documento 22
2 FUNDAMENTAÇÃO DA PESQUISA 23
2.1 Requisito de Software 23
2.2 GORE – Goal Oriented Requirements Engineering 27
2.3 Elementos de Valor (Elements of Value) 32
2.4 PBSRS – Problem Based Software Requirements Specification 35
2.5 Histórias de Usuário 39
2.6 Discussão sobre o Capítulo 41
3.1 Considerações Adicionais sobre o Domínio das Necessidades 43
3.1.1 Necessidades dos Clientes 43
3.1.2 Categorização de Necessidades dos Clientes 45
3.2 Conceituação de Ontologia 47
3.3 Proposta de Ontologia para Requisitos de Software 48
3.4 Conceituação de Stakeholder 51
3.5 Conceituação de Domínio dos Problemas 53
3.6 Conceituação de Domínio das Necessidades 57
3.7 Conceituação de Domínio dos Requisitos 61
3.8 Discussão sobre o Capítulo 64
4 DESCRIÇÃO DA TÉCNICA PROPOSTA 66
4.1 Visão Geral da Estruturação da Associação entre os Domínios 66
4.1.1 Estruturação de Domínios de Especificação e Associações 66
4.1.2 Associação entre os Domínios 68
4.1.3 Análise das Associações entre Domínios 70
4.2 Técnica de Especificação de Necessidades Proposta 72
4.2.1 Visão Geral da Especificação de Necessidades do Cliente 72
4.2.2 Descrição da Técnica Proposta 77
4.2.3 Exemplo de Aplicação da Técnica Proposta 79

5 CASO DE ESTUDO 83
5.1 Visão Geral do Negócio 83
5.1.1 Visão Geral do Contexto de Negócio 83
5.1.2 Definição dos Stakeholders 84
5.2 Especificação de Problemas 85
5.3 Glance 86
5.4 Especificação das Faltas 87
5.5 Especificação dos Entregáveis 88
5.6 Especificação de Necessidades 88
5.7 Análise de Coberturas dos Problemas 89
5.8 Especificação de Requisitos 90
5.9 Análise de Cobertura das Necessidades 92
5.10 Discussão sobre o Caso de Estudo 93
6 DISCUSSÃO 95
6.1 Consistência com Contexto de Especificação de Requisitos 95
6.2 Contribuição Técnica dos Conceitos de Necessidades 96
6.3 Consistência dos Conceitos com Outras Abordagens 97
6.3.1 GORE – Goal Oriented Requirements Engineering 97
6.3.2 Elementos de Valor (Elements of Value) 98
6.3.3 História de Usuário 100
7 CONCLUSÃO 102
REFERÊNCIAS 104
ANEXO A - Tesauro 109

1 INTRODUÇÃO

Este capítulo apresenta uma introdução ao trabalho de pesquisa de mestrado, indicando o contexto da pesquisa, o tema escolhido, os problemas abordados e questões de pesquisa, além das motivações, objetivos e organização do documento.

1.1 Contexto de Pesquisa

O trabalho de pesquisa proposto se insere na problemática geral de Requisitos de Software. Trata-se de um assunto amplo, impactante e de grande interesse acadêmico e profissional, que vem sendo estudado há décadas, visando estabelecer formas eficazes de realização. Entretanto, a atividade de Requisitos de Software em desenvolvimento de software ainda é deficiente em muitas, senão na maioria, das empresas, o que motiva novos estudos e aprofundamentos (TENBERGEN e DAUN, 2019) (FERNÁNDEZ, et al., 2017) (OKESOLA, et al., 2019).

Dentro deste assunto de Requisitos de Software, esta pesquisa focou no tema da Especificação das Necessidades dos Clientes e seu relacionamento com a especificação dos requisitos de software, que corresponde a uma das etapas dentro da abordagem de especificação denominada *Problem Based Software Requirements Specification* (PBSRS), desenvolvida na Universidade Tecnológica Federal do Paraná (UTFPR). Esta especificação de necessidades determina o que o software deverá prover para o cliente e, por isso, é anterior à especificação dos requisitos do software (*i.e.*, requisitos funcionais e não funcionais) e representa a determinação da “causa” ou o “porquê”, que leva aos requisitos exigidos do software (SOUZA e STADZISZ, 2016).

A atividade de Requisitos de Software é uma das integrantes dos Processos de Desenvolvimento de Software. Estes processos são estudados e propostos dentro da área de conhecimento denominada Engenharia de Software, criada em 1968 e que envolve os conhecimentos requeridos para o desenvolvimento de software com o objetivo de obter a qualidade necessária, alta produtividade e, principalmente, a garantia de que o produto atenda as demandas dos clientes (NAUR e RANDELL, 1969). Segundo o *Software Engineering Body Of Knowledge* (SWEBOK), a Engenharia de Software é “uma abordagem sistemática aplicada,

disciplinada e quantificável para o desenvolvimento, a operação e a manutenção de software" (BOURQUE e FAIRLEY, 2014).

A Engenharia de Software é uma área de conhecimento e de atuação profissional de muito grande relevância na atualidade, pois provê instrumentos para melhoria da eficiência, eficácia e qualidade no desenvolvimento de software (ASSYNE, et al., 2022). Dada a gigantesca demanda mundial por software na maioria (senão quase todos) os setores de atividade e a crescente dependência da sociedade por software, o impacto da Engenharia de Software, também, cresce continuamente (ASSYNE, et al., 2022) (GIRAY, 2021).

Efetivamente, software é uma peça central do desenvolvimento tecnológico, uma vez que uma grande proporção das novas tecnologias e inovações incorporam softwares que têm o papel de conduzir a lógica dos sistemas em termos de execução das funcionalidades, monitoramento das ações, controle das operações, interconexão com outros sistemas, segurança e tomadas de decisões mais inteligentes, entre outras (CICO, et al., 2021). Desta forma, ter domínio em desenvolvimento de software e, portanto, em Engenharia de Software, é um diferencial significativo para a competitividade nas empresas (ASSYNE, et al., 2022) (GIRAY, 2021) (CICO, et al., 2021).

Isto posto, há cinco atividades fundamentais envolvidas na Engenharia de Software que são (BOURQUE e FAIRLEY, 2014): Requisitos de Software (*i.e.*, *Software Requirements*), Projeto do Software (*i.e.*, *Software Design*), Construção do Software (*i.e.*, *Software Construction*), Teste de Software (*i.e.*, *Software Testing*) e Manutenção de Software (*i.e.*, *Software Maintenance*). A atividade de Requisitos de Software tem um papel fundamental nesta área, já que é nesta atividade que se realiza o entendimento das exigências dos clientes sobre o que o futuro software deverá fazer e ter como propriedades. Por sua vez, o projeto envolve a concepção da solução que será implementada (*i.e.*, arquitetura, componentes, interfaces e outras características) para atender aos requisitos especificados. A seguir, realiza-se a construção ou implementação do software de acordo com seu projeto. O software produzido é, então, verificado por meio de testes e outras técnicas para checar sua qualidade e o atendimento aos requisitos do cliente. Por fim, ocorre a manutenção que são atualizações do software devido a defeitos identificados, mudanças de operações e a novos requisitos dos clientes, dentre outros.

Como dito anteriormente, a atividade de Requisitos de Software é a atividade da Engenharia de Software em que são ou deveriam ser coletadas informações sobre as demandas ou intenções dos clientes e especificados, de forma precisa, correta e completa, os requisitos que guiarão o desenvolvimento de software. Pode-se dizer, então, que a atividade de Requisitos de Software é crítica, pois o trabalho realizado está diretamente relacionado ao sucesso ou fracasso do software em atender as demandas dos clientes. Caso esta atividade não seja realizada de forma adequada, pode-se alcançar um produto que não seja atrativo ou útil aos clientes e aumenta-se, assim, o risco de desinteresse dos clientes pelo software. Desta forma, as exigências dos clientes, e demais *stakeholders*, devem ser bem entendidas e descritas nas especificações de requisitos para que sejam bem compreendidas por todos os envolvidos (*i.e.*, cliente e time de desenvolvimento) e que levem, ao final, a um produto útil ao cliente.

Para melhorar a qualidade da especificação, a atividade de Requisitos de Software inclui, classicamente, as etapas de elicitação, de análise, de especificação e de validação de requisitos (BOURQUE e FAIRLEY, 2014). Na elicitação, obtém-se informações necessárias dos *stakeholders* em relação ao software para que seja feita a análise da solicitação, da viabilidade, dos conflitos dos requisitos e do limite do projeto. Após esta análise, os requisitos são especificados, ou seja, é criado um documento que contém o que é esperado do software na forma de restrições, especificações das exigências dos *stakeholders* de forma mais técnica, os custos, o prazo, dentre outras informações importantes do projeto analisado. A validação ocorre após a criação do documento de especificação de requisitos de software, com todos os envolvidos, para verificarem se os requisitos levantados estão completos e corretos.

A comunicação efetiva é, também, um grande desafio, já que os profissionais envolvidos (*i.e.*, engenheiro, analista e/ou especialista em requisitos) devem entender as intenções dos *stakeholders*, coletando informações dos clientes de acordo com sua experiência e conhecimento do domínio de negócio em diferentes níveis da organização, bem como lidar com conflitos e mudanças de exigências. Assim, a atividade de requisitos de software deve prover meios para auxiliar a comunicação e o entendimento entre os profissionais de desenvolvimento de software e os *stakeholders*.

Diante disto, percebe-se que há uma dificuldade de associar as intenções que motivam o cliente a procurar um software com o que é especificado nos requisitos. Em geral, na visão da autora e da PBSRS (SOUZA e STADZISZ, 2016), os empreendimentos de desenvolvimento de software não investem em detalhar as intenções dos clientes e demais *stakeholders*, de forma que elas ficam, quando muito, subentendidas e abstratas. Outras vezes, busca-se coletar os requisitos diretamente com os *stakeholders*, entendendo (erroneamente) que as funcionalidades e características do software representam as intenções dos *stakeholders*.

Embora haja no mercado ferramentas de rastreabilidade de requisitos, com as quais é possível associar os requisitos definidos com as fontes que levaram a estes requisitos, isto não garante que a especificação esteja completa e correta. Estas ferramentas auxiliam na associação dos requisitos a suas fontes, que são documentos gerais ou outros registros que indicam a origem das exigências do cliente, como atas de reuniões, histórias de usuários (*i.e.*, *user stories*), questionários e outros levantamentos realizados. Entretanto, a definição das fontes dos requisitos não define, precisamente, uma relação causal e sim uma relação de origem, ou seja, pode-se concluir que um requisito foi criado por uma demanda do cliente ocorrida em determinado momento da elicitação de requisitos, mas não se pode saber o porquê da demanda (*i.e.*, a sua causa ou as intenções que levam aos requisitos). Sem o conhecimento das causas dos requisitos não é possível saber se estes estão corretos e completos e, portanto, se as funcionalidades e propriedades do software, ao final, contemplarão os interesses dos *stakeholders*.

Por isso, percebe-se que existe uma grande lacuna entre as demandas de negócios (*i.e.*, intenções dos *stakeholders* com o software) e a especificação de requisitos de software. Para entender as reais demandas dos *stakeholders*, é necessário haver uma análise mais elaborada e ampla, de modo que o analista de requisitos consiga verificar se o software realmente conseguirá prover aquilo que se deseja na perspectiva dos clientes.

Embora a especificação de requisitos seja uma atividade que, sabidamente, integra os processos de desenvolvimento de software e sobre a qual são e foram desenvolvidas pesquisas ao longo de várias décadas, as abordagens, métodos e ferramentas atuais ainda carecem de efetividade. A visão clássica da atividade de requisitos de software (BOURQUE e FAIRLEY, 2014) pode ser considerada

bastante genérica e abstrata, dando orientações gerais, mas sem prover conceitos precisos e técnicas sistemáticas que possam auxiliar efetivamente o analista de requisitos. A visão clássica colabora muito mais na organização do trabalho de especificação de requisitos do que na sua realização. O SWEBOK, por exemplo, no seu capítulo 3, sobre a Elicitação de Requisitos, apresenta conceitos vagos sobre o que deveria ser alcançado (*e.g.*, problemas, objetivos, requisitos técnicos e de negócios, casos de uso, histórias de usuário) e, também, apresenta técnicas alternativas nesta etapa, mas que não parecem ser consistentes entre elas, entre outras, pela própria terminologia emprega.

Outras abordagens mais recentes para especificação de requisitos, trazem alguma clareza e fundamentação melhores, como é o caso do GORE (GIORGINI, et al., 2005) e da PBSRS (SOUZA e STADZISZ, 2016). Particularmente, esta última foi proposta na UTFPR, pelo grupo de pesquisa ao qual este trabalho de mestrado está associado. Estas abordagens serão mais bem discutidas no próximo capítulo deste documento.

1.2 Problemas e Questões de Pesquisa

A pesquisa de mestrado apresentada neste documento considera os problemas relatados a seguir, dentro do contexto discutido nesta introdução. O Problema de Base configura a dificuldade maior à qual esta pesquisa se direciona. Os problemas específicos, por sua vez, configuram as dificuldades particulares de interesse desta pesquisa e das quais se desdobram as três questões de pesquisa, relacionadas na sequência.

Problema de Base

P1 - Não raramente, produtos de software fracassam em alcançar sucesso (*e.g.*, vendas suficientes para justificar o negócio), pois não são considerados úteis (*i.e.*, em termos de funcionalidades oferecidas, benefícios e valor) ou viáveis (*i.e.*, relação custo-benefício e prazo) pelo mercado (*i.e.*, conjunto de clientes potenciais). Seguramente, este é um problema gravíssimo, pois leva à falência do produto e a perdas no investimento feito no seu desenvolvimento.

Problemas Específicos

P2 - O especialista em requisitos tem dificuldade em estabelecer o conjunto de requisitos para um software que seja completo (*i.e.*, não faltem requisitos frente às demandas do cliente) e correto (*i.e.*, não tenha requisitos desnecessários às demandas do cliente). Este problema advém do fato de que, em abordagens clássicas de especificação, as fontes de informação para a atividade de requisitos de software não apresentam uma semântica clara para os conceitos usados, nem alguma técnica mais precisa e compreensível para a elaboração da especificação.

P3 - O especialista em requisitos tem dificuldade em entender a real demanda dos clientes, a partir da qual elaborar os requisitos, pois muitas fontes de informação são vagas ou incompletas, além da caracterização das demandas não ser clara.

Questões de Pesquisa

Q1 - O conceito de “Necessidades do Cliente”, proposto por Rafael M. de Souza (SOUZA e STADZISZ, 2016), é consistente dentro do contexto da especificação de requisitos de software e pode auxiliar na solução dos problemas P2 e P3, e, consequente, P1?

Q2 - A introdução de conceitos de “Necessidades do Cliente” pode trazer contribuição técnica para a atividade de requisito de software e pode auxiliar na solução dos problemas P2 e P3, e, consequente, P1?

Q3 - A aplicação do conceito de “Necessidades do Cliente” é consistente com os conceitos de “objetivos” do GORE, com os conceitos de “valor” do *Elements of Value* e com os de Histórias de Usuário e pode auxiliar na solução dos problemas P2 e P3, e, consequente, P1?

Este trabalho de mestrado visa prover respostas às questões Q1, Q2 e Q3, contribuindo para a evolução da abordagem PBSRS em desenvolvimento na UTFPR. As motivações e objetivos da pesquisa são descritos nas seções a seguir.

1.3 Motivações para a Pesquisa

Desde a criação da Engenharia de Software em 1968 (NAUR e RANDELL, 1969), quando se caracterizou a “crise de software”, tem-se indicadores de que, frequentemente, os softwares desenvolvidos não atendem as exigências dos clientes. Ao longo das décadas seguintes, mesmo com as evoluções tecnológicas observadas, este problema não foi resolvido, nem amenizado. O “*The Chaos Report*” (STANDISH, 1995) reforça esta observação, colocando as falhas na especificação de requisitos entre as três principais razões para o insucesso e cancelamento de desenvolvimentos de software. Estudos mais recentes corroboram esta mesma observação (TENBERGEN e DAUN, 2019) (FERNÁNDEZ, et al., 2017).

A abordagem “clássica” proposta para a especificação de requisitos (conforme será discutido na seção 2.1, segundo o SWEBOK) apresenta uma visão padrão de como conduzir a atividade de Requisitos de Software, mas que é pouco efetiva dada sua superficialidade. Seus conceitos são limitados e as técnicas são generalistas e pouco consistentes com os conceitos. Além disso, esta abordagem clássica é pouco sistemática, possui lacunas que dificultam a compreensão e não orientam adequadamente o especialista em requisitos no que deveria ser feito para se alcançar uma “boa especificação”. Por exemplo, diz-se que a especificação de requisitos deverá ser “completa”, ou seja, diz-se que não deverá ser “esquecido” nenhum requisito importante do cliente. Porém, a abordagem não fornece nenhum meio claro para alcançar este critério.

Embora outras abordagens mais efetivas tenham sido propostas (*e.g.*, GORE), a atividade de especificação, ainda, é um grande desafio em todos os desenvolvimentos de software e, ainda, são necessárias abordagens mais efetivas para aumentar a qualidade das especificações e, consequentemente, aumentar a taxa de sucesso dos desenvolvimentos de software.

Neste contexto, a motivação geral deste trabalho é contribuir para o aprimoramento da abordagem de especificação de requisitos de software denominada PBSRS, em desenvolvimento da UTFPR. Esta contribuição estará focada no aprofundamento do conceito de Domínio das Necessidades e sua aplicação na atividade de requisitos.

A relevância desta pesquisa está em colaborar para o aumento da taxa de sucesso de desenvolvimentos de software, podendo ter efeitos econômicos positivos

na indústria de software. Ademais, a possibilidade de aumento da qualidade das especificações de requisitos, também, colabora para a maior competitividade das empresas de software e inovação tecnológica.

1.4 Objetivo da Pesquisa

O objetivo geral deste trabalho de pesquisa de mestrado é contribuir com o aperfeiçoamento da abordagem PBSRS desenvolvida na UTFPR por meio do detalhamento dos conceitos e da proposição de uma técnica de especificação das “Necessidades dos Clientes” e de seu relacionamento com os “Requisitos de Software”.

Os objetivos específicos da pesquisa de mestrado são:

1. Aprofundar os conceitos relacionados com o Domínio das Necessidades dos Clientes, visando torná-los mais precisos e práticos aos profissionais, incluindo a definição de uma ontologia de requisitos de acordo com a abordagem PBSRS.

2. Rever e detalhar a técnica proposta na abordagem PBSRS para a especificação das Necessidades dos Clientes, uma vez que a especificação proposta até então é, ainda, informal.

3. Realizar um caso de estudo de aplicação real da técnica proposta e dos conceitos em desenvolvimentos de software.

4. Realizar uma análise crítica da aplicação dos conceitos e da técnica proposta a partir do caso de estudo realizado, visando caracterizar sua utilidade dentro da abordagem PBSRS e responder as questões de pesquisa Q1, Q2 e Q3.

1.5 Método de Pesquisa Empregado

O método de pesquisa empregado neste trabalho de mestrado envolveu sete etapas, conforme descrito a seguir:

Etapa 1 – Definição dos Objetivos da Pesquisa

Nesta etapa, fez-se um estudo dos trabalhos de pesquisa anteriores e em andamento (notadamente a PBSRS) no PPGCA e CPGEI/UTFPR a respeito de especificação de requisitos de software e definiu-se o tema desta pesquisa de mestrado. A seguir, formulou-se os problemas e questões de pesquisa e definiu-se,

então, o objetivo geral e objetivos específicos que orientaram os trabalhos deste mestrado.

Etapa 2 – Estudo Bibliográfico e da Fundação da Pesquisa

Nesta pesquisa, buscou-se uma compreensão do estado da arte e da técnica a respeito da atividade de requisitos de software por meio da leitura de uma coleção de trabalhos científicos selecionados. A partir deste estudo bibliográfico, pode-se estabelecer a fundamentação da pesquisa desenvolvida.

Etapa 3 – Estudo da Estruturação da Associação entre os Domínios

Nesta etapa, analisou-se e complementou-se a estruturação de domínios de especificações e a associação entre eles. Além disso, foi realizado uma análise das associações entre domínios de acordo com a abordagem PBSRS, apoiada na teoria de projeto axiomático (SUH, 1990).

Etapa 4 – Aprofundamento da Conceituação sobre Necessidades do Cliente e Proposição de uma Ontologia para Especificação de Requisitos de Software

Nesta etapa, foi realizada uma análise aprofundada dos conceitos e da semântica utilizados para descrever o Domínio das Necessidades dos Clientes. Em seguida, propôs-se uma ontologia de maneira a proporcionar uma melhor precisão nas definições dos conceitos associados aos *Stakeholder*, Domínio dos Problemas, Domínio das Necessidades e Domínio dos Requisitos e suas relações.

Etapa 5 – Proposição de uma Técnica de Especificação de Necessidades

Nesta etapa, propôs-se uma técnica visando auxiliar o especialista na elaboração da especificação das necessidades dos clientes, empregando os conceitos do Domínio das Necessidades do Cliente e baseando-se nos trabalhos anteriores em PBSRS.

Etapa 6 - Realização de Caso de Estudo

Nesta etapa, foi realizado um caso de estudo para aplicar, tecnicamente, os conceitos propostos, visando obter um diagnóstico sobre sua utilização. O caso de

estudo abordou um sistema real de *marketplace* em desenvolvimento na empresa na qual a estudante atuava.

Etapa 7 - Análise dos Resultados Alcançados

Nesta etapa, foi realizado um estudo dos resultados alcançados com a aplicação da técnica e conceitos propostos no caso de estudo. O objetivo foi produzir uma análise crítica sobre a pesquisa e as proposições.

1.6 Estrutura do Documento

Este documento está organizado em sete capítulos, incluindo este da introdução. O Capítulo 2 apresenta as fundamentações do trabalho apresentando uma visão geral da atividade de Requisitos de Software especificação de requisitos em uma perspectiva clássica e discorre, também, sobre a abordagem orientada a objetivos (GORE), sobre Elementos de Valor, sobre Histórias de Usuários e sobre a abordagem PBSRS desenvolvida na UTFPR. O Capítulo 3 apresenta o aprofundamento da conceituação sobre o domínio das necessidades dos clientes e a proposta de uma ontologia para requisitos de software segundo a PBSRS e a conceituação dos domínios desta proposta. O Capítulo 4 apresenta a técnica proposta para especificação de necessidades, incluindo o estudo da estruturação de domínios de especificação e associações. O Capítulo 5 apresenta um caso de estudo sobre a especificação de necessidades e requisitos, empregando a técnica proposta. O Capítulo 6 presenta discussões sobre a pesquisa e, por fim, o Capítulo 7 apresenta as conclusões do trabalho.

2 FUNDAMENTAÇÃO DA PESQUISA

Neste capítulo serão apresentadas as referências de conhecimento que fundamentam a pesquisa proposta nesta dissertação. Em especial, serão discutidos os conceitos fundamentais da atividade de Requisitos de Software e serão apresentadas as abordagens GORE, Elementos de Valor, Histórias de Usuários e PBSRS.

2.1 Requisito de Software

A atividade de Requisitos de Software é uma importante atividade da Engenharia de Software (TENBERGEN e DAUN, 2019) (FERNÁNDEZ, et al., 2017) (SOUZA e STADZISZ, 2016), uma vez que é responsável por determinar e especificar tecnicamente as exigências dos *stakeholders*, em termos do que o futuro software deverá fazer (*i.e.*, requisitos funcionais) e ter como características ou restrições (*i.e.*, requisitos não funcionais).

O SWEBOK, uma das principais referências da Engenharia de Software produzido pela *Institute of Electrical and Electronics Engineers* (IEEE) - *Computer Society*, descreve uma abordagem clássica para especificação de requisitos, consistente com a maioria das publicações acadêmicas e científicas sobre especificação de requisitos (BOURQUE e FAIRLEY, 2014). Esta abordagem de especificação envolve etapas que segem.

Elicitação de Requisitos

A elicitação de requisitos é a etapa da atividade de Requisitos de Software em que o especialista em requisitos busca compreender (*i.e.*, também referido como coletar, adquirir, descobrir, identificar ou elicitar) as fontes dos requisitos, o conhecimento do domínio e as exigências dos *stakeholders* em relação ao software a ser desenvolvido (BOURQUE e FAIRLEY, 2014) (LIM, et al., 2021) (LI, et al., 2020).

Várias técnicas podem ser utilizadas nesta etapa para auxiliar a identificação dos requisitos, entre elas: (i) entrevistas, (ii) cenários, (iii) protótipos, (iv) reunião facilitada, (v) observação e (vi) histórias de usuários (BOURQUE e FAIRLEY, 2014). A entrevista é a técnica mais utilizada na elicitação de requisitos, porque permite

uma interação direta do especialista em requisitos com os *stakeholders* (LI, et al., 2020). A definição de cenários é utilizada para entender a interação do usuário com o sistema em diferentes situações (OKESOLA, et al., 2019) e pode tomar a forma de um modelo de casos de uso, por exemplo. Já o protótipo, é uma técnica visual e de experimentação visando explorar as demandas e exigências dos *stakeholders* (OKESOLA, et al., 2019). Pode-se realizar a “reunião facilitada” com um grupo ou todos os *stakeholders* para obter, em conjunto, requisitos mais consistentes (OKESOLA, et al., 2019) uma vez que a análise conjunta do grupo poderia melhor identificar inconsistências entre os requisitos. Emprega-se, também, a “observação” para aprofundar o entendimento de situações mais complexas, acompanhando como o usuário executa suas atividades no cotidiano (OKESOLA, et al., 2019). Por fim, a “história de usuário” é uma técnica popular para especificar requisitos de alto nível em que se escreve o que os *stakeholders* querem que o software faça para alcançar algum valor de negócio ou benefício (BOURQUE e FAIRLEY, 2014) (TRKMAN, et al., 2019) (LUCASSEN, et al., 2016) (COHN, 2004).

Deve-se considerar, também, que a experiência e o conhecimento do especialista em requisitos sobre o domínio do negócio poderão afetar a efetividade na utilização das técnicas de elicitação, influenciando a sua capacidade de entendimento da demanda e do escopo do projeto (LI, et al., 2020) (BAGHERI, et al., 2019). Não é raro que a comunicação dos especialistas em requisitos com os *stakeholders* seja inefetiva (SOUZA e STADZISZ, 2016), por razões como a diferença de formação profissional (*e.g.*, tipo de conhecimento e a experiência nos processos de negócios e tecnologias), a diferença cultural (*e.g.*, nível de maturidade) e a diferença de papéis (*i.e.*, o especialista em requisitos deve buscar entender a demanda que é do *stakeholder*). Estas questões dificultam o entendimento adequado das demandas dos *stakeholders*, que se agravam quando os *stakeholders* ainda ajustam ou mudam suas exigências ao longo do desenvolvimento do software (FERNÁNDEZ, et al., 2017) (SOUZA e STADZISZ, 2016).

Análise de Requisitos

Após elicitar os requisitos, realiza-se a etapa de análise na qual os requisitos são classificados, modelados conceitualmente, alocados dentro da arquitetura do software, negociados e documentados (BOURQUE e FAIRLEY, 2014).

Os requisitos são classificados em requisitos funcionais e não funcionais, em tipos de requisitos (*i.e.*, de processo ou de produto), em prioridades, em volatilidade (*i.e.*, possibilidade de os requisitos virem a ser modificados) e em nível do requisito (*i.e.*, requisito de alto nível e requisito de baixo nível) (BOURQUE e FAIRLEY, 2014) (SINGH, et al, 2016) (NAVARRO-ALMANZA, et al., 2017) (ABD ELWAHAB, et al, 2016). Também, é realizada a decomposição dos requisitos de alto nível e a verificação dos requisitos que estão dentro do escopo do projeto (NAVARRO-ALMANZA, et al., 2017). Na modelagem conceitual, por sua vez, elabora-se um modelo abstrato da interação dos requisitos da organização com os requisitos operacionais, empregando diagramas (*e.g.*, diagrama de caso de uso, modelo de estado, modelo de dados) (LAPLANTE, et al., 2022). Em seguida, realiza-se a alocação de requisitos aos elementos da arquitetura da solução de forma a indicar as reponsabilidades dos componentes do software com o conjunto de requisitos (BOURQUE e FAIRLEY, 2014). Na negociação entre as partes interessadas, são solucionados os conflitos entre os requisitos, priorizados os requisitos, elaborado o cronograma de entrega, feita a análise de riscos e valor do projeto (ALVES, et al., 2010). Após a negociação, os resultados são aprovados com todos os envolvidos (ALVES, et al., 2010).

Especificação de Requisitos de Software

A especificação de requisitos, geralmente, se refere à produção de até três tipos de documentos de especificação: (i) definição de sistema, (ii) requisitos do sistema e (iii) requisitos de software, que podem ser sistematicamente revisados, avaliados e aprovados pelo cliente e contratante ou fornecedor (BOURQUE e FAIRLEY, 2014). No entanto, a produção desses documentos irá depender da complexidade e necessidades do desenvolvimento do software, sendo que o documento de especificação de requisitos de software é o mais comumente utilizado nos projetos (BOURQUE e FAIRLEY, 2014).

O documento de definição do sistema e o de requisitos do usuário definem uma lista de requisitos de alto nível de acordo com o domínio do usuário, que traduzem as exigências e as restrições do futuro software (ALVES, et al., 2010). Eles possuem, também, informações dos *stakeholders* envolvidos, os cenários de uso e os fluxos de trabalho (ALVES, et al., 2010) (SCHÖN, et al., 2017). Podem, ainda, conter ilustrações para auxiliar o entendimento do atual sistema e a expectativa

sobre o futuro software em relação aos negócios da empresa (ALVES, et al., 2010) (SCHÖN, et al., 2017).

Por fim, o documento de especificação de requisitos de software possui, ainda, informações do escopo do projeto e a análise para o desenvolvimento do software (*e.g.*, a estimativa de custo, a análise de riscos, o cronograma de entrega, requisitos funcionais e requisitos não funcionais), documentando estas especificações com as partes envolvidas (BOURQUE e FAIRLEY, 2014). As especificações de requisitos de software são escritas, mais comumente, em linguagem natural, de forma que devem ser entendidas por todos os envolvidos. No entanto, a qualidade dos requisitos de software é um dos problemas apontados nesta atividade, ou seja, os requisitos devem ser escritos com uma semântica clara, sem redundâncias, com legibilidade e com rastreabilidade para com a real demanda do cliente.

Validação

Ao finalizar o documento de especificação de requisitos, ao menos, um representante do time de desenvolvimento e do cliente devem validar este documento para garantir que os requisitos especificados atendam as intenções dos *stakeholders* em relação ao futuro software e verificar se o documento está de acordo com os padrões da empresa e se os requisitos estão claros, consistentes, completos e corretos (BOURQUE e FAIRLEY, 2014) (KAMALRUDIN e SIDEK, 2015) (IEEE, 1998).

Algumas técnicas que auxiliam na execução desta etapa são: (i) revisão de requisitos, (ii) prototipagem, (iii) validação do modelo e (iv) teste de aceitação (BOURQUE e FAIRLEY, 2014). A revisão de requisitos é uma técnica em que um grupo de revisores fazem uma inspeção dos requisitos definidos, visando localizar erros, suposições inadequadas, falta de clareza e desvios de práticas padrões (KAMALRUDIN e SIDEK, 2015). Por sua vez, a técnica de prototipagem permite uma validação por meio da construção simplificada do software ou parte específica dele para que os *stakeholders* experimentem e interajam com os requisitos que foram especificados, validando, assim, se os requisitos foram compreendidos corretamente e se o software poderá atender a sua demanda (SCHÖN, et al., 2017). Também, pode-se efetuar a validação da qualidade do modelo conceitual que foi realizado na etapa de análise, por meio de diferentes técnicas, inclusive

matemáticas (*e.g.*, equações matemáticas e dados de modelagem) (INAYAT, et al., 2015) (THACKER, et al., 2004). Por fim, existe a técnica de criar testes de aceitação para os *stakeholders*, que validam se os requisitos especificados estão atendendo sua real demanda (DALTON, 2019).

Para finalizar, no entendimento do SWEBOK, os “Requisitos de software expressam as necessidades e restrições atribuídas a um produto de software que contribuem para a solução de algum problema do mundo real” (BOURQUE e FAIRLEY, 2014). Nota-se que nesta interpretação, os requisitos se equivalem a “necessidades”, pois seriam a sua expressão. Os requisitos se relacionam diretamente com a resolução de problemas, porém não é feita referência a nenhum elemento causal dos requisitos. A resolução de problemas indica o propósito ou motivação final para buscar por um software, mas não determina, suficientemente, o porquê de cada requisito especificado.

2.2 GORE – Goal Oriented Requirements Engineering

Uma abordagem para a especificação de requisitos bem referenciada e disseminada na literatura é a abordagem orientada a objetivos (*i.e.*, *goals*), denominada *Goal Oriented Requirements Engineering* (GORE) (*i.e.*, Engenharia de Requisitos Orientada a Objetivos) (HORKOFF, et al., 2019) (ALJAHDALI, et al., 2011). Nesta abordagem, a especificação de requisitos é estendida além dos requisitos, considerando os chamados “objetivos” (ALJAHDALI, et al., 2011). Entende-se que todos os requisitos do software se desdobram de um conceito mais abstrato expresso na forma de um ou mais objetivos. Dito de outra forma, entende-se que um software deverá realizar certas funções para atender a determinado objetivo do cliente ou do negócio. Assim, os objetivos representariam as intenções finais do cliente e a especificação dos requisitos se originaria deles (HORKOFF, et al., 2019) (ALJAHDALI, et al., 2011).

Nesta abordagem, especificam-se os objetivos a serem alcançados em alto nível em duas etapas denominadas *Early Requirements Analysis* e *Late Requirements Analysis* (GIORGINI, et al., 2005). A seguir, a partir dos objetivos definidos nestas duas etapas, desenvolve-se a especificação dos requisitos visando atender a estes objetivos.

Na *Early Requirements Analysis* são identificados os objetivos vinculados aos *stakeholders*, juntamente com as suas tarefas e recursos (HORKOFF e YU,

2016). Nesta etapa, preocupa-se em realizar o mapeamento no contexto mais organizacional para deixar claro os “porquês” das motivações do cliente em relação ao sistema e como este sistema irá funcionar neste contexto (HORKOFF e YU, 2016). Este mapeamento é representado por um diagrama de ator que apresenta os atores envolvidos e as dependências entre eles, denominadas “acordos” entre os atores que possuem um objetivo em comum (GIORGINI, et al., 2005). Este objetivo poderá ser um *softgoals*, um objetivo, uma tarefa ou um recurso. A diferença entre *softgoal* e objetivo é a definição clara de critérios do objetivo a ser cumprido, ou seja, quando a definição não está tão clara considera-se “um *softgoal*” e quando a definição está melhor, considera-se “um objetivo”. A representação de cada item do diagrama é mostrada na Tabela I (GIORGINI, et al., 2005).

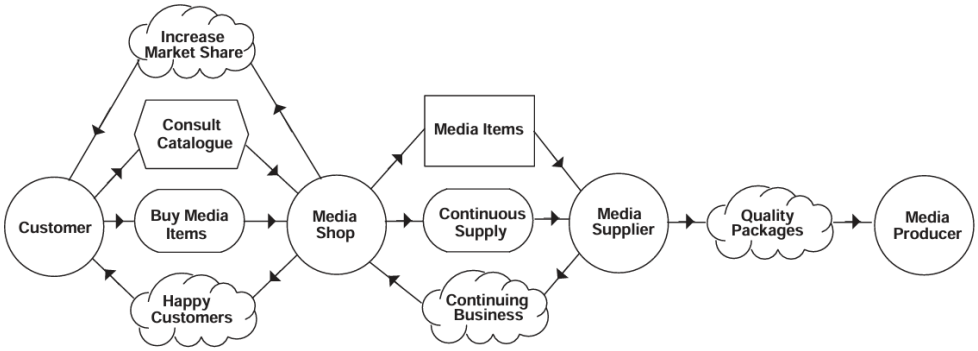
Tabela I - Representação dos itens do diagrama

Item	Representação
Ator	
Softgoal	
Objetivo	
Tarefa	
Recurso	

Fonte: P. Giorgini, J. Mylopoulos e R. Sebastini (2005)

Para exemplificar, um “diagrama de ator” de um sistema de vendas digital, cujos atores são *Customer*, *Media Shop*, *Media Supplier* e *Media Producer*, é apresentado na Figura 1. Nota-se que o sistema “*Media Shop*” possui quatro objetivos relacionados com o consumidor (*i.e.*, *Customer*). O *Media Shop* depende do consumidor para atingir dois objetivos abstratos (*i.e.*, *Softgoals*) que são o de aumentar sua participação no mercado (*e.g.*, *Increase Market Share*) e tornar os clientes felizes (*i.e.*, *Happy Customers*). Além disso, o consumidor depende do *Media Shop* para atingir dois objetivos que são consultar o catálogo de produto (*i.e.*, *Consult Catalogue*) e comprar itens de mídia (*i.e.*, *Buy Media Items*) (GIORGINI, et al., 2005).

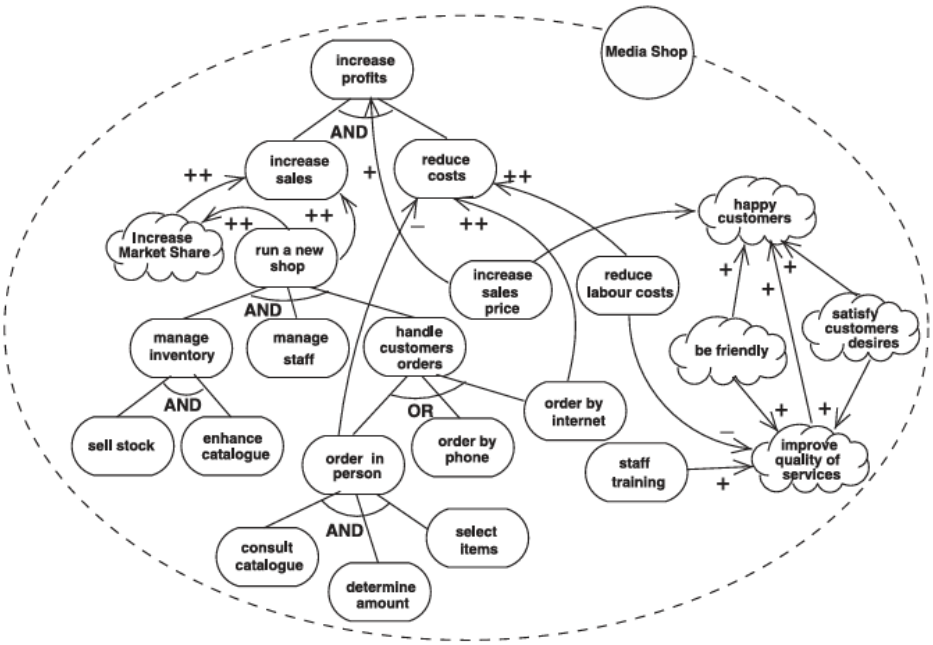
Figura 1 - Diagrama de ator do *Media Shop*



Fonte: P. Giogini, J. Mylopoulos e R. Sebastini (2005)

Adicionalmente, o *Media Shop* depende do fornecedor de mídias (*i.e.*, *Media Supplier*) para atingir o objetivo de fornecimento contínuo (*i.e.*, *Continous Supply*). Este, por sua vez, depende do *Media Shop* para atingir seu objetivo de continuidade dos negócios (*i.e.*, *Continuing Business*) e depende dos fabricantes (*i.e.*, *Media Producer*) para alcançar o objetivo abstrato de qualidade de empacotamento (*i.e.*, *Quality Packages*) (GIORGINI, et al., 2005).

Figura 2 - Diagrama de explicação do ator *Media Shop*



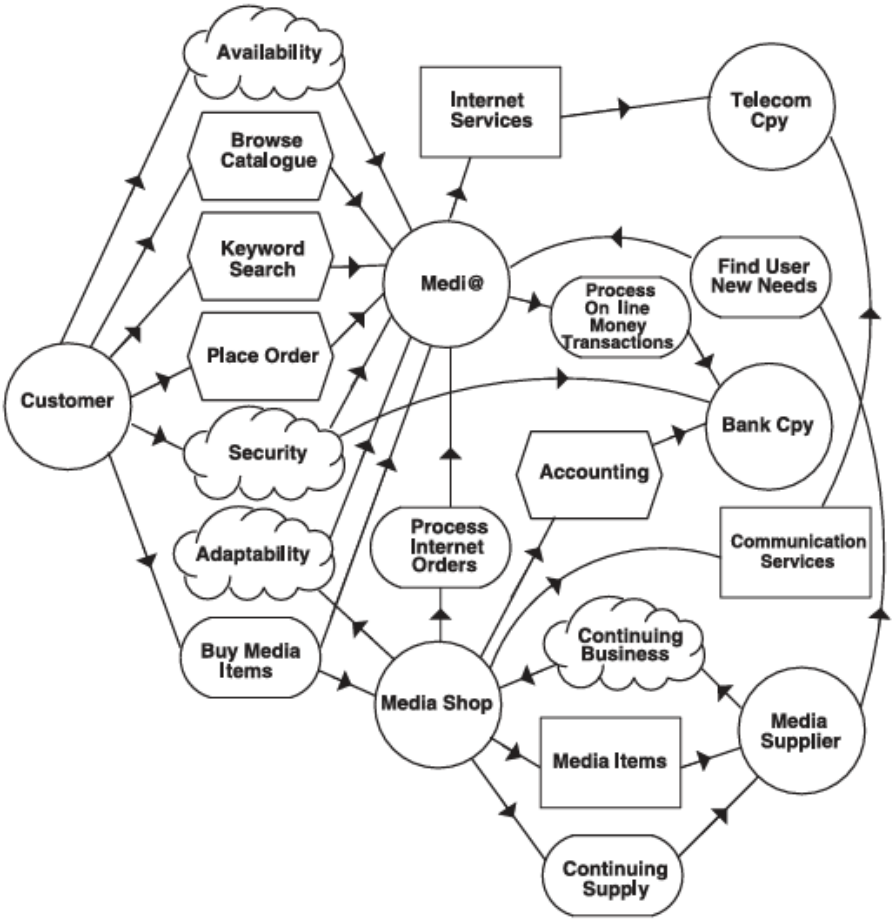
Fonte: P. Giogini, J. Mylopoulos e R. Sebastini (2005)

Para realizar uma análise e detalhamento de requisitos em GORE, cria-se um Diagrama de Explicações (*i.e.*, *Rationale Diagram*) que irá detalhar e descrever

decomposições dos objetivos e dependências com outros atores. A Figura 2 apresenta o diagrama de explicações do ator *Media Shop*, em que os objetivos deste ator são refinados (*i.e.*, decompostos), indicando se os objetivos (ou subobjetivos) contribuem de forma positiva ou negativa para atingir os objetivos maiores. Por exemplo, o *softgoal* “satisfazer os desejos do consumidor” (*i.e.*, *satisfy customers desires*) gera um resultado positivo para o *softgoal* “tornar os clientes felizes” (*i.e.*, *Happy Customers*) (GIORGINI, et al., 2005).

Na etapa de *Late Requirements Analysis* são identificadas as exigências dos clientes em relação ao software (*e.g.*, requisitos funcionais e requisitos não funcionais), conforme ilustrado na Figura 3 (GIORGINI, et al., 2005).

Figura 3 - Diagrama de ator do *Media Shop*

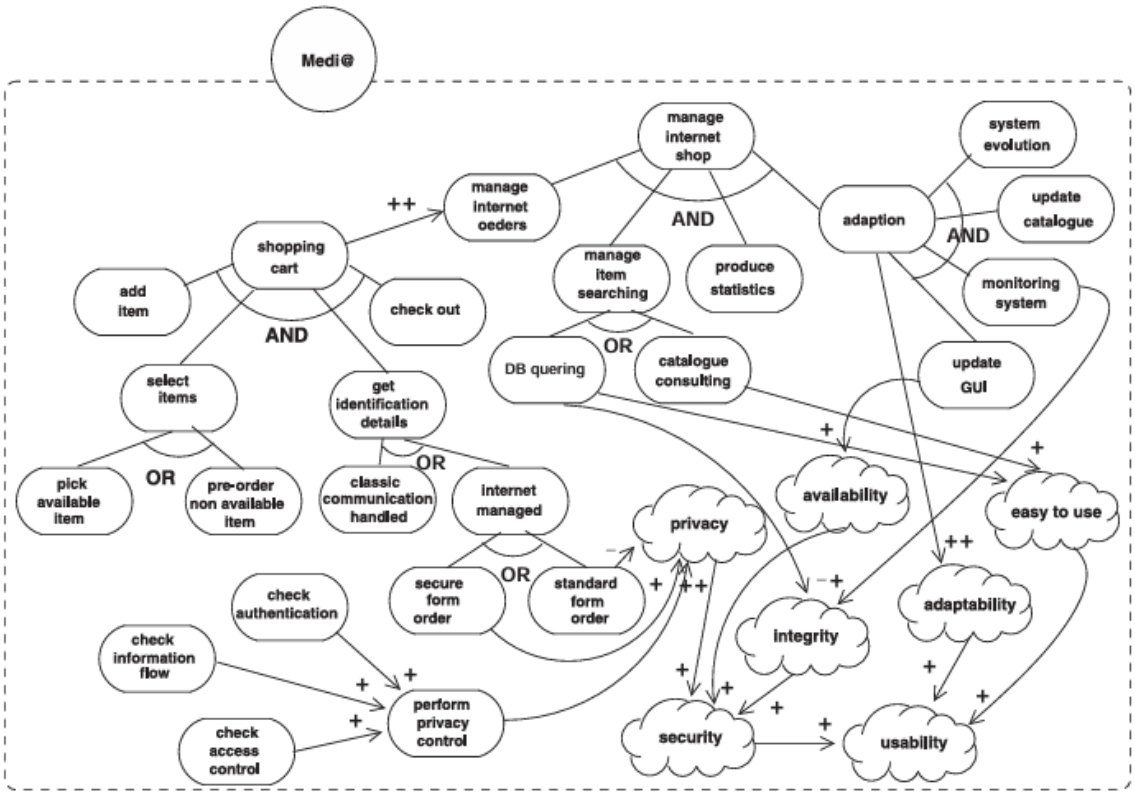


Fonte: P. Giogini, J. Mylopoulos e R. Sebastini (2005)

Na Figura 1, percebe-se que para o *Media Shop* aumentar a sua participação no mercado ou tornar clientes felizes houve a inclusão de três atores na

Figura 3, que são: *Medi@*, *Bank Cpy* e *Telecom Cpy* e, consequentemente, mais itens para alcançar os objetivos do *Media Shop*. Ou seja, o *Customer* depende do *Medi@* para consultar o catálogo (*i.e.*, *Browse Catalogue*), buscar por palavras-chave (*i.e.*, *Keyword Search*) e fazer pedidos de compra (*i.e.*, *Place Order*). A Figura 3 apresenta, ainda, os demais objetivos relacionados, incluindo os *softgoals*. Por sua vez, na Figura 4 apresenta-se mais um nível de detalhamento dos objetivos do *Medi@* na forma de decomposições em subobjetivos (GIORGINI, et al., 2005).

Figura 4 - Diagrama de explicação para o *Media Shop*



Fonte: P. Giogini, J. Mylopoulos e R. Sebastini (2005)

A abordagem GORE tem gerado muitos estudos científicos derivados, pois trouxe para a Engenharia de Requisitos um elemento novo, explicitamente considerado, que é a especificação das causas dos requisitos na forma de objetivos (ALJAHDALI, et al., 2011) (HORKOFF e YU, 2016) (ROLLAND e SALINESI, 2005). Assim, esta abordagem contribui para reduzir o distanciamento que havia entre a especificação técnica dos requisitos do produto e a intenção original dos clientes (HORKOFF, et al., 2019) (ALJAHDALI, et al., 2011).

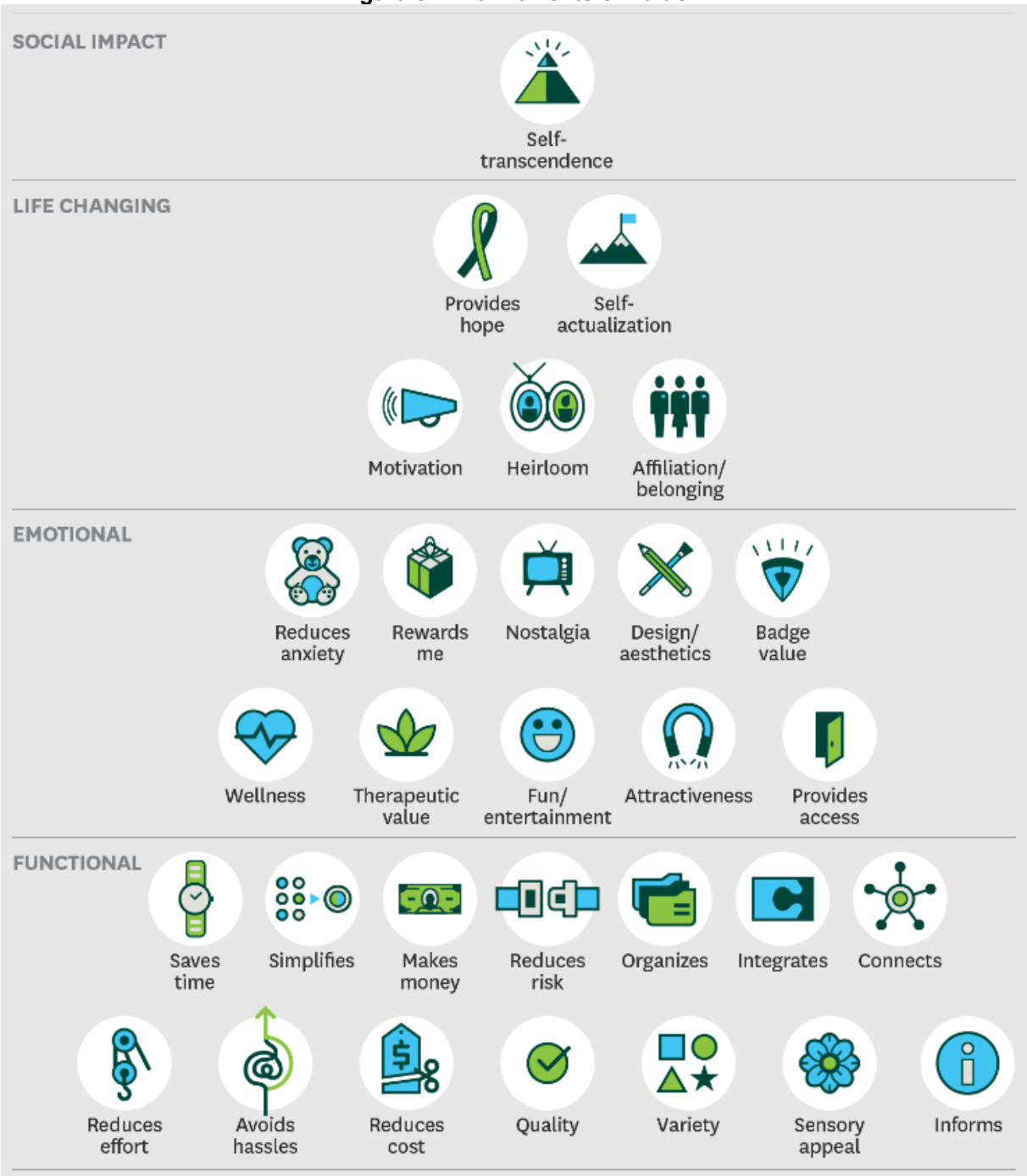
Contudo, a contribuição da abordagem GORE é limitada e, mesmo, o conceito fundamental de “objetivo” é impreciso. Exatamente, pela imprecisão semântica do termo, “objetivos” são comumente confundidos com “requisitos funcionais de alto nível”. A própria definição de “*goal*”, já revela esta má formulação do termo: “*Functional goals underlie services that the system is expected to deliver whereas non-functional goals refer to expected system qualities*” (*i.e.*, os objetivos funcionais estão subjacentes aos serviços que se espera que o sistema forneça, enquanto os objetivos não funcionais referem-se às qualidades esperadas do sistema) (ROLLAND e SALINESI, 2005). Ou seja, o termo “*goal*” está diretamente vinculado às funções e características dos produtos, mesmo que de forma mais abstrata (VAN LAMSWEERDE, 2004). Emprega-se, ainda, o termo “*softgoal*” que poderia estar mais ligado aos interesses do negócio em que produto seria utilizado. Mas, afirma-se que a satisfação de tais objetivos não poderia ser estabelecida de forma precisa (LIM, et al., 2021) e que são usados mais para comparações.

Pode-se considerar que GORE tem sido uma contribuição importante para que a especificação de requisitos avance em busca de um mapeamento mais adequado de requisitos, porém sua contribuição é bastante incompleta e imprecisa, mesmo que tenha tido uma popularidade respeitável (ALJAHDALI, et al., 2011) (FERNÁNDEZ, et al., 2017). A viabilidade e utilidade do GORE nas empresas ainda não foram evidenciadas e, assim, não se sabe que se esta abordagem é robusta e prática o suficiente para aplicações reais (MAVIN, et al., 2017).

2.3 Elementos de Valor (Elements of Value)

O conceito de “valor” é popularmente utilizado nas empresas, principalmente pelo setor de *marketing*, para designar os “benefícios percebidos” pelos clientes dos produtos ou serviços ofertados (GRIGG, 2021). Afirma-se, também, que o valor percebido é uma relação entre o que o produto entrega de benefícios frente a seu custo. Assim, diz-se que as empresas precisam se esforçar para “entregar valor” para o consumidor como uma das estratégias aplicadas para se destacarem da concorrência, aumentarem a satisfação do cliente, definirem a precificação, se comunicarem de forma assertiva com os consumidores e, até mesmo, criarem produtos ou serviços (GRIGG, 2021) (ALMQUIST, et al., 2016).

Figura 5 - *The Elements of Value*



Fonte: E. Almquist, J. Senior e N. Bloch (2016)

Em um trabalho recente de destaque, E. Almquist, J. Senior e N. Bloch propuseram uma classificação de "valores" denominada "Elements of Value" (ALMQUIST, et al., 2016). Esta classificação tem a forma de uma pirâmide organizada em quatro categorias separando quatro tipos de valores: Funcional (*i.e.*, *Functional*), Emocional (*i.e.*, *Emotional*), Mudança de Vida (*i.e.*, *Life Changing*) e

Impacto Social (*i.e.*, *Social Impact*), conforme ilustrado na Figura 5. São propostos trinta elementos de valor de forma que, em geral, quanto maior o número e intensidade de elementos de valor fornecidos pelo produto ao cliente, maior será a lealdade dos clientes, maior será o *Net Promoted Score* (*i.e.*, grau de satisfação dos clientes) e maior será o crescimento do faturamento sustentável (ALMQUIST, et al., 2016).

A classificação de "*Elements of Value*" tem como origem os conceitos da hierarquia das necessidades de Maslow, na qual são classificadas as necessidades humanas. Assim como na pirâmide de necessidades, a classificação de "*Elements of Value*" representa, na base da pirâmide, os valores mais básicos ou mais concretos, enquanto aqueles mais ao topo representam valores mais complexos ou abstratos (ALMQUIST, et al., 2016). Da mesma forma, a consideração destes elementos de valor pode permitir a redução da falha na especificação e desenvolvimento de novos produtos e em processos de inovação (GRIGG, 2021).

Nas proposições sobre "*Elements of Value*", afirma-se que os elementos de valor "endereçam" as "necessidades" ou "aspirações" dos clientes, e que se inspiram nos conceitos da Pirâmide de Necessidades de Maslow (ALMQUIST, et al., 2016). Assim, entende-se que os "valores" não são as "necessidades", mas sim o que o produto oferece em vista das necessidades dos clientes. Cada elemento de valor pode ser visto como uma categoria ou classe de valor, visando satisfazer uma categoria ou classe de necessidade. Por exemplo, quando se faz referência ao elemento de valor "*Quality*", no sentido de que o produto fornece uma percepção de qualidade ao cliente, se está fazendo referência a uma classe de valor, pois a qualidade ofertada pode assumir e ser percebida de diferentes formas, como a qualidade aparente (*i.e.*, estética), a qualidade de materiais componentes, a qualidade de acabamento (*e.g.*, precisão) ou, ainda, a qualidade em termos de poucos defeitos de funcionamento. Por outro lado, o termo "necessidade" é usado pelos autores de "*Elements of Value*", mas não é esclarecido. Entende-se que os autores utilizam este termo no sentido comum, de entendimento evidente, e em analogia com aquele usado por Maslow.

No capítulo 3 será feita uma discussão mais aprofundada a respeito do uso do termo "necessidades dos clientes". Este termo tem relevância para esta pesquisa de mestrado e assumirá uma interpretação mais detalhada e fundamentada. Nas

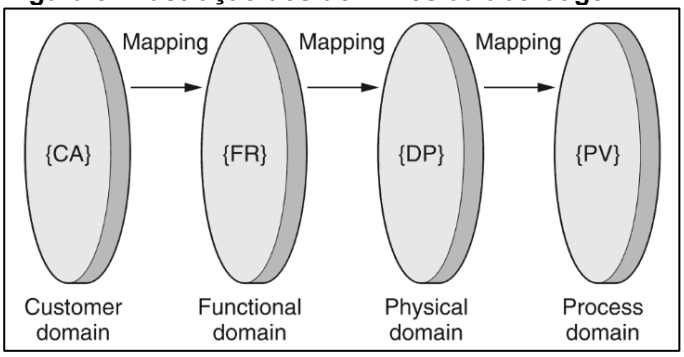
discussões finais do Capítulo 6 será feito uma associação de conceitos de necessidades e de valor.

2.4 PBSRS – Problem Based Software Requirements Specification

A abordagem PBSRS está em desenvolvimento na UTFPR (SOUZA e STADZISZ, 2016), e propõe um conjunto de conceitos e um mapeamento entre domínios, visando permitir a especificação de requisitos de software de forma mais sistemática, a partir da identificação dos problemas de negócio do cliente. Esta organização do conceito de mapeamento entre os domínios originou-se nos fundamentos da abordagem denominada *Axiomatic Design* (*i.e.*, Princípios Axiomáticos) desenvolvida por Nam Suh no *Massachusetts Institute of Technology* (MIT), que segue uma cadeia de dependências causais em projetos (SUH, 1990).

Como ilustrado na figura a seguir, Suh (2005) propõe quatro domínios de conhecimento envolvidos em um desenvolvimento de projeto na área de mecânica ou de engenharia de produção que servem como linha de demarcação entre os conhecimentos dos quatro tipos de atividades em um projeto. O Domínio do Cliente (*i.e.*, *Customer Domain*) é caracterizado por Atributos (*i.e.*, CAs – *Customers Attributes*) que o cliente está buscando em um produto (ou processo, material, sistema, organização). No Domínio Funcional (*i.e.*, *Functional Domain*), os CAs são especificados em termos de Requisitos Funcionais (*i.e.*, FRs - *Funcional Requirements*) e Restrições (*i.e.*, *Constraints*). Para satisfazer as FRs especificadas, concebem-se Parâmetros de Projeto (*i.e.*, DP – *Design Parameters*) no Domínio Físico (*i.e.*, *Physical Domain*). Por fim, para produzir o produto especificado em termos dos DPs, desenvolve-se um processo que é caracterizado pela Variáveis de Processo (*i.e.*, PVs - *Process Variables*) no Domínio de Processo (*i.e.*, *Process Domain*).

Figura 6 - Ilustração dos domínios da abordagem AD



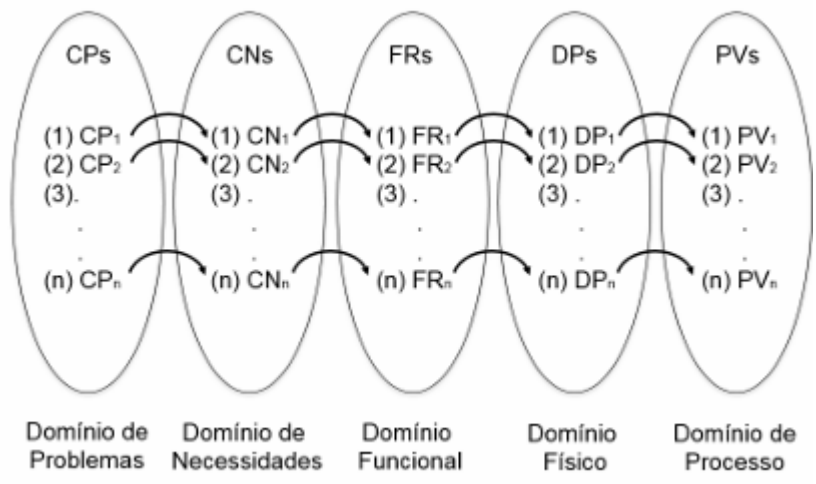
Fonte: Suh (2005)

Os mapeamentos entre os domínios (*i.e.*, *Mapping*), ilustrados na Figura 6, indicam relações entre “o que” (domínio da esquerda) e “como” (domínio da direita). Ou seja, representam mapeamentos entre as exigências e a solução. Assim, estes mapeamentos estabelecem relações de causa e efeito, no sentido de que os domínios à esquerda, que descrevem “o que se quer alcançar”, representam as causas para os domínios à direita, que descrevem “como alcançar o que foi especificado”, ou seja, são os efeitos.

A partir desta definição de domínios e do mapeamento entre as suas variáveis, Suh (1990) estabelece a teoria de *Axiomatic Design*, envolvendo dois axiomas, nove corolários e dezesseis teoremas. Os dois axiomas estabelecem os fundamentos da teoria que são os conceitos de independência funcional e conteúdo de informação (ou complexidade). Estes axiomas orientam o projetista no desenvolvimento de soluções otimizadas para as demandas dos clientes.

Na abordagem PBSRS (SOUZA, 2016), adotou uma representação de domínio ligeiramente adaptada e expandida daquela de Suh (1990), conforme ilustrado na Figura 7. A abordagem PBSRS fragmenta o Domínio do Cliente, da abordagem *Axiomatic Design*, em Domínio dos Problemas e Domínio das Necessidades, visando refinar o entendimento da causa que faz o cliente buscar por um software (*i.e.*, problema) e o que este produto deverá entregar (*i.e.*, necessidade) para o cliente diante deste problema (PEREIRA, 2011). Os demais domínios (Funcional, Físico e Processo) possuem as mesmas características do *Axiomatic Design*, que são os conhecimentos clássicos relacionados com a especificação funcional (*i.e.*, dos requisitos funcionais e não funcionais), especificação técnica da solução projetada e com a implantação desta solução.

Figura 7 - Ilustração dos domínios da abordagem PBSRS

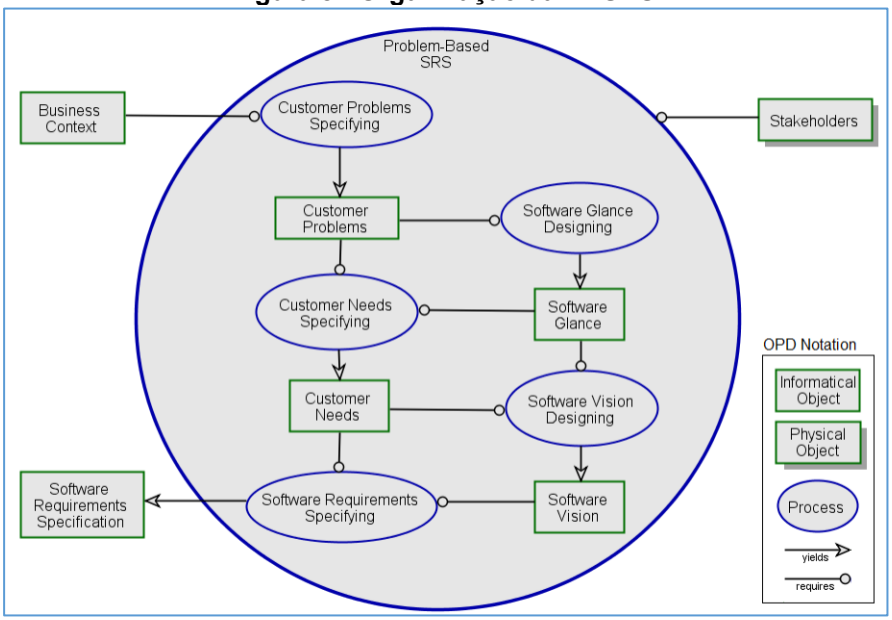


Fonte: Souza (2016)

A abordagem PBSRS destaca três domínios de especificação que são: o Domínio de Problemas, o Domínio de Necessidades e o Domínio Funcional. Os conhecimentos dentro destes domínios têm uma relação causal que deve ser mapeada de forma que se possa avaliar a qualidade dos requisitos produzidos, em especial se a especificação está correta e completa ao final (PEREIRA, 2011). Neste sentido, uma especificação será considerada “correta” se todos os requisitos estão mapeados a necessidades e problemas dos clientes e será considerada “completa” se todos os problemas estão mapeados a necessidades e requisitos do software (SOUZA, 2016).

A organização da abordagem PBSRS está apresentada na Figura 8, utilizando a notação da *Object Process Methodology* (OPM), na qual as elipses em azul representam processos e os retângulos em verde representam objetos (SOUZA e STADZISZ, 2016).

Figura 8 - Organização da PBSRS



Fonte: R. Souza e P. Stadzisz (2016)

A aplicação da PBSRS inicia pelo processo de Especificação dos Problemas do Cliente (*i.e.*, *Customer Problems Specifying*) que determina os motivos ou causas que levam o cliente a buscar uma solução, na forma de um ou mais problemas. Em geral, um problema representa algo negativo ou alguma dificuldade relacionada com o contexto de negócios do cliente e que gera uma motivação de busca por uma solução. A partir dos problemas dos clientes, elabora-se uma idealização ou vislumbre de como um software poderia resolver estes problemas (*i.e.*, *Software Glance Designing*), no que os autores denominam *glance* (*i.e.*, vislumbre ou esboço da solução). Trata-se da primeira e mais abstrata descrição da solução de software. A seguir, realiza-se a especificação “do que” é preciso que o software entregue ou produza para resolver os problemas, denominada Especificação das Necessidades do Cliente (*i.e.*, *Customer Needs Specifying*). Sabendo-se o que é esperado que o software proveja (*i.e.*, *Customer Needs*), pode-se refinar o *glance* da solução (*i.e.*, *Software Vision Designing*) gerando o documento visão do software (*i.e.*, *Software Vision*). Por fim, especifica-se “o que” o software deverá fazer e ter como propriedades (*i.e.*, *Software Requirements Specifying*), na forma de requisitos, visando compor uma solução adequada aos problemas apontados pelo cliente (SOUZA e STADZISZ, 2016).

Assim, considera-se, nesta abordagem, que um produto é útil e traz satisfação ao cliente quando o que ele entrega resolve ou auxilia a resolver problemas deste cliente. Dito de outra maneira, considera-se que um produto de sucesso é aquele que é útil ao cliente na resolução de seus problemas. Desta forma, a PBSRS propõe que se parta do entendimento e especificação dos problemas para, então, derivar as necessidades e requisitos, avaliando-se, a cada passo, o mapeamento (*i.e.*, associação) entre os domínios (SOUZA e STADZISZ, 2016).

Além disso, como ilustrado na Figura 8, a abordagem PBSRS pode ser realizada de forma sequencial, concluindo-se cada uma das 5 etapas uma por vez ou, ainda, pode ser realizada de forma iterativa por meio de ciclos de refinamento (*i.e.*, modelo incremental em amplitude ou em profundidade) (SOUZA e STADZISZ, 2016). Neste sentido, é possível, inclusive, empregar a PBSRS iterativa integrada a algum método ágil.

2.5 Histórias de Usuário

As Histórias de Usuário (*i.e.*, *User Story* ou *Stories*) são uma técnica originalmente proposta por Kent Beck (1999), durante o desenvolvimento do projeto *Chrysler* C3 em *Detroid*, em 1997. Esta técnica envolve a descrição informal e em linguagem natural dos recursos (*i.e.*, *features*) desejados do software, sob a perspectiva do usuário (*i.e.*, do *stakeholder*), e tornou-se um dos fundamentos da abordagem denominada Programação Extrema (*i.e.*, *Extreme Programming* – XP). Segundo Kent Beck (1999), Histórias de Usuário são escritas pelos usuários e cada uma envolve uma porção de um ou mais casos de uso e será desenvolvida só ou em conjunto com outras histórias em uma iteração no processo de desenvolvimento. As histórias são escritas em texto livre e curto sobre um cartão de anotação (*i.e.*, *Note Card*). Um exemplo de história de usuário seria: “Um usuário pode postar seu currículo no *website*” (*i.e.*, *A user can post her resume to the website*). (COHN, 2004).

Mike Cohn (2004), em seu livro “*User Stories Applied: For Agile Software Development*”, popularizou a técnica de Histórias de Usuário apresentando o modelo Connextra. Esse modelo foi inventado pela equipe de desenvolvimento da empresa Connextra, que foi uma das primeiras adotantes da abordagem XP. O modelo incorporou o “papel dos usuários” nas histórias e definiu um padrão (*i.e.*, *template*) para escrita das histórias. Este padrão tem a forma: “Eu como um (papel) quero

(função) de modo que (valor de negócio) (*i.e.*, "*I as a (role) want (function) so that (business value)*").

Atualmente, o padrão mais utilizado pelos profissionais para escrita de histórias de usuário é: "Como um (papel), eu quero (objetivo), para que (benefício)" (*i.e.*, *As a <role>, I want < goal> so that < benefit>*). O "papel" descreve um tipo de usuário, ou seja, uma classificação da responsabilidade ou função de um *stakeholder*, como usuário, administrador ou coordenador. O "objetivo" descreve a ação que o usuário quer que o software execute em seu suporte. Por fim, o "benefício" fornece a razão para esta ação para o usuário, ou seja, é o porquê ele quer o objetivo apontado (AMNA e POELS, 2022). Um exemplo de história de usuário deste padrão é "Como cliente, quero transferir fundos entre minhas contas vinculadas, para poder financiar meu cartão de crédito" (*i.e.*, "*As a customer, I want to transfer funds between my linked accounts, so that I can fund my credit card*") (AMNA e POELS, 2022).

As Histórias de Usuário são bastante populares atualmente, sendo a forma preferida de expressão de requisitos em métodos ágeis (*e.g.*, Scrum, XP). Segundo (COHN, 2004), as histórias de usuário auxiliam na colaboração de todos os envolvidos (equipe de desenvolvimento, cliente e demais *stakeholders*) para entender as perspectivas dos clientes em relação ao que será desenvolvido e entregue em cada iteração (COHN, 2004). Lucassen et al (2016) enfatiza que a vantagem do uso das histórias de usuário é o conhecimento do "benefício" que a "função" (*i.e.*, papel) terá ao atingir o "objetivo".

Os maiores problemas apontados desta técnica são a existência de ambiguidades e conflitos entre as histórias, considerando as perspectivas dos diferentes *stakeholders*, ou seja, as histórias podem não ser claras, consistentes e precisas (AMNA e POELS, 2022) (WANG, et al., 2023).

Na visão da autora desta dissertação, é possível notar que estas dificuldades com as histórias de usuário se originam, principalmente, na imprecisão semântica desta técnica. No exemplo de Amna e Poels (2022), percebe-se que o "objetivo" especificado envolve uma ação que o software deverá fazer, como "transferir fundos entre minhas contas vinculadas", ou seja, trata-se de um requisito funcional e não um objetivo. Além disso, o "benefício" também pode ser considerado um objetivo, como "para financiar um cartão", pois retrata o que se deseja alcançar.

2.6 Discussão sobre o Capítulo

Como visto neste capítulo de fundamentação da pesquisa, o campo de conhecimento de Requisitos de Software tem um impacto muito grave dentro dos processos de desenvolvimento de software. Isto se deve ao fato de que a atividade de Requisitos de Software contribui decisivamente no sucesso dos produtos, influenciando diretamente o atendimento das necessidades e a satisfação dos clientes.

Para bem conduzir a atividade de Requisitos de Software, técnicas foram estudadas e propostas ao longo dos anos. Uma técnica ou abordagem bem consolidada é apresentada no SWEBOK e pode ser considerada como a abordagem clássica neste campo de conhecimento, integrando um grande conjunto de propostas anteriores sobre Requisitos de Software. Outras técnicas mais recentes, como GORE, *Elements of Value*, PBSRS e Histórias de Usuário, buscam ir mais longe nos conceitos e práticas da atividade de Requisitos de Software, conforme reportado neste capítulo.

Um aspecto observado durante a pesquisa foi a falta de clareza sobre o conceito de origem dos requisitos. Em uma visão mais analítica, não é difícil entender que os requisitos para um software não são um fim e sim um meio. Ou seja, os clientes não adquirem um software pelo que ele faz, nem por suas características, mesmo que eles considerem estes elementos na decisão pela compra. Como apontado na abordagem PBRSRS, mas também em outras fontes, há uma similaridade sobre o fato de os requisitos terem uma causa e ela ser advinda de uma motivação dos clientes. É esta motivação que leva os clientes a desejarem adquirir um software, considerando a sua potencial contribuição ou utilidade para atender a uma demanda percebida pelo cliente. O termo mais comum encontrado na literatura para denominar essa causa dos requisitos é “necessidades dos clientes” (BOURQUE e FAIRLEY, 2014) (GIORGINI, et al., 2005) (SUH, 1990).

Com vista a contribuir para melhorar a qualidade da atividade de Requisitos de Software, desenvolve-se na UTFPR um trabalho de pesquisa para proposição de uma abordagem denominada PBSRS. Com esta abordagem, pretende-se alcançar uma conceituação com semânticas mais precisas, de forma que o entendimento e especificação dos requisitos possam ser mais efetivos. A PBSRS propõe, também, uma técnica para especificação envolvendo dois domínios adicionais ao domínio dos

requisitos de forma a permitir que o especialista possa desenvolver a especificação de maneira mais sistemática e compreensível. Essa abordagem é útil para esclarecer a causa dos requisitos, que estão intrinsicamente ligados às necessidades, as quais, por sua vez, são originadas de problemas (SOUZA e STADZISZ, 2016).

Este trabalho de pesquisa de mestrado busca trazer contribuições à abordagem PBSRS, focando no Domínio das Necessidades dos Clientes e sua associação com a especificação de requisitos de software.

3 ONTOLOGIA PARA REQUISITOS DE SOFTWARE

Este capítulo apresenta a conceituação sobre Necessidades do Cliente, que é o objeto desta pesquisa, e descreve uma proposta de ontologia para auxiliar a atividade de Requisito de Software de acordo com a abordagem PBSRS. Em especial, são destacados os conceitos de *Stakeholder* e dos domínios do Problema, das Necessidades e dos Requisitos.

3.1 Considerações Adicionais sobre o Domínio das Necessidades

Discute-se, nesta seção, a interpretação de Necessidades do Cliente que é um dos conceitos centrais dentro da abordagem PBSRS, estendendo os trabalhos anteriores do grupo de pesquisa (SOUZA e STADZISZ, 2016) (PEREIRA, 2011).

3.1.1 Necessidades dos Clientes

Muitos estudos, tanto na área de *marketing* (BAYUS e SHANE, 2008) (GERACIE e EPPINGER, 2013) quando em engenharia (SOUZA e STADZISZ, 2016) (BOURQUE e FAIRLEY, 2014) (GIORGINI, et al., 2005) (ALJAHDALI, et al., 2011) (SUH, 1990) (GROEN, et al., 2017) (INCOSE, 2015) (MORANDINI, et al, 2017), têm buscado entender a visão dos *stakeholders* (*i.e.*, *The Voice of Customers*) a respeito de seus propósitos, antes de se estabelecer os requisitos para os produtos, de forma que os requisitos reflitam aquilo que os clientes “esperam” que os produtos lhes entreguem. Na literatura científica, considera-se que os requisitos para um produto/software se originam de “demandas” dos clientes (GERACIE e EPPINGER, 2013), porém não há um consenso sobre a melhor terminologia, nem técnica, para descrever tal “demanda” dos clientes e, muito frequentemente, ela é confundida com as próprias exigências técnicas e funcionais sobre o produto em si (BOURQUE e FAIRLEY, 2014) (GIORGINI, et al., 2005) (SEBOK, 2023).

Considere-se, como exemplo, que uma empresa expressa sua intenção de adquirir um software para gerar um determinado relatório mensal. Assim, seria possível entender que “gerar um relatório” seria a “demanda” ou “propósito” do *stakeholder* (*i.e.*, da empresa cliente). Entretanto, na verdade, “gerar um relatório” é a exigência funcional que recairá sobre o software e não uma real demanda ou

propósito. De fato, nenhum software representa um objetivo por si só e sim um meio ou ferramenta para se alcançar alguma demanda ou propósito dos *stakeholders* (SOUZA e STADZISZ, 2016). Desta forma, o que realmente interessa aos *stakeholders* não são os produtos ou serviços oferecidos e sim o que eles produzem, no que resultam ou o que entregam. Nestes termos, pode-se afirmar que:

“Os clientes não precisam dos produtos ou serviços propriamente ditos, mas sim do que estes lhe entregam”.

Falhas na correta e completa definição de requisitos em virtude de não se alcançar um entendimento claro das demandas do cliente, pode, certamente, colocar em risco o sucesso de um produto (LI, et al., 2020). Quando se trata de inovação, por exemplo, as taxas de falha (*i.e.*, insucesso) de novos produtos podem, comumente, alcançar 90%, conforme citações na imprensa popular e acadêmica, que, também, sugerem que “a inovação bem-sucedida é a exceção e não a regra” (BAYUS e SHANE, 2008).

Nesta pesquisa, propõe-se separar, claramente, a expressão da “demanda dos clientes” e a expressão dos requisitos do produto/software, permitindo que se estabeleça os relacionamentos entre estes dois domínios. Com relação ao domínio da “demanda dos clientes” (ou, simplesmente, “Domínio do Cliente”), encontram-se muitas formas de expressão, fazendo referências aos “desejos dos clientes”, “entrega de valor” (ALMQUIST, et al., 2016), “objetivos de negócio” (GIORGINI, et al., 2005) (ALJAHDALI, et al., 2011) (MORANDINI, et al, 2017), “benefícios aos clientes”, “expectativas do consumidor”, “necessidades do cliente” (SOUZA e STADZISZ, 2016) (INCOSE, 2015), entre outros. Estas distintas formas de expressão criam dificuldades em se entender mais precisamente qual é o objeto que bem descreve a “demanda dos clientes” (FERNÁNDEZ, et al., 2017). Em todo o caso, em suma, o termo mais comumente encontrado na literatura e nas práticas profissionais para se fazer referência aos conhecimentos no Domínio do Cliente é “Necessidades do Cliente” (*i.e.*, em inglês Customer´s Needs) (SOUZA e STADZISZ, 2016) (BOURQUE e FAIRLEY, 2014) (GROEN, et al., 2017) (BAYUS e SHANE, 2008) (INCOSE, 2015).

Uma “necessidade” de um cliente seria algo essencial para ele (MICHAELIS, 2024), que, inclusive, o levaria a buscar um produto para suprir esta necessidade.

Além disso, o termo “necessidade” também remete à “percepção de falta ou carência de algo” (MICHAELIS, 2024). Neste sentido, uma necessidade seria algo relevante que o cliente percebe que não possui e do qual sente a falta. Por exemplo, quando se afirma “eu necessito me alimentar ou nutrir.”, isso significa que o indivíduo sente ou percebe uma carência de nutrientes ou falta de alimentação em seu organismo. Também, quando um gerente financeiro afirma “Eu preciso saber quando efetuar o pagamento dos tributos mensais.”, ele está declarando que lhe falta o conhecimento para o correto pagamento dos tributos devidos.

O sentimento de falta é o que caracteriza a necessidade do cliente e que faz com que o mesmo busque por “algo” (*e.g.*, um produto ou um serviço) que possa lhe trazer a satisfação desejada. Pode-se dizer, também, que as necessidades do cliente representam os motivos ou razões que o levam a buscar um produto e que, portanto, são “as causas ou os porquês” dos requisitos (SOUZA e STADZISZ, 2016).

Outros termos são também empregados para expressar as “necessidades do cliente” usando verbos como: precisar, querer, almejar, objetivar, desejar e buscar (SOUZA e STADZISZ, 2016). Todos eles apontam a mesma condição de motivação de se buscar algo que não se tem e do qual se sente falta. Por exemplo, quando um cliente afirma “eu quero saber quais foram as vendas do dia.”, ele está indicando que lhe falta o conhecimento sobre as vendas e que esta falta é relevante ao ponto de ele buscar um meio de supri-la. Assim, “querer saber”, “desejar saber” ou “necessitar saber” são semanticamente equivalentes. O mesmo vale para os outros termos citados.

A partir das considerações feitas nesta seção (MICHAELIS, 2024) (CAMBRIDGE, 2022) (SOUZA e STADZISZ, 2016), define-se o termo “necessidade do cliente” da seguinte forma:

‘‘Uma necessidade de um cliente é a especificação de algo significativo do qual o cliente sente falta e que seria provido pelo produto almejado.’’

3.1.2 Categorização de Necessidades dos Clientes

As necessidades do cliente podem ser expressas de maneiras variadas e podem ser categorizadas para auxiliar seu entendimento e suas especificações (SOUZA e STADZISZ, 2016) (BAYUS e SHANE, 2008).

O sentimento de falta que caracteriza uma necessidade pode variar em frequência (BAYUS e SHANE, 2008), podendo ser categorizado em contínuo, recorrente ou ocasional. Uma necessidade contínua nunca se esgota e deveria ser atendida incessantemente. Um exemplo seria a necessidade de estabilização em voo de um avião que deveria ser provida por algum sistema de controle da aeronave. Por outro lado, uma necessidade recorrente pode ser atendida por um determinado produto, mas cuja falta poderia voltar a se manifestar posteriormente. Um exemplo seria a necessidade de irrigação de uma plantação, que exigiria um sistema de acionamento sempre que a umidade do solo estivesse abaixo de um certo patamar. Ainda, uma necessidade poderia ser categorizada como ocasional, na situação em que a manifestação da falta não é regular. Um exemplo seria a necessidade do cliente de ter conhecimento de seu saldo bancário.

As necessidades do cliente podem, também, variar em intensidade, já que nem todo sentimento de falta faz com que o cliente tenha a mesma motivação de busca de uma solução. Na verdade, a intensidade de uma necessidade deriva da gravidade do problema e de suas penalizações, ao qual ela se associa (SOUZA e STADZISZ, 2016). Assim, se um problema tiver consequências (*i.e.*, penalizações) graves ao cliente, as necessidades associadas a ele tenderão a ter alta intensidade (SOUZA e STADZISZ, 2016). Por exemplo, a “obrigação de pagar os funcionários até o último dia útil de cada mês” é um problema grave para as empresas, pois as penalizações podem envolver multas e processos trabalhistas. Assim, uma necessidade como “precisar de um sistema de informação para saber quando efetuar os pagamentos mensais” pode ser considerada “intensa” dada a gravidade do problema associado.

Bayus e Shane (2008) diferenciam as necessidades do cliente em Articuladas e Não Articuladas. As necessidades articuladas são aquelas expressas diretamente ou verbalizadas pelo cliente, se questionado adequadamente por meio de entrevistas, grupos focais ou questionários, entre outras técnicas de pesquisa e elicitação. As necessidades Não Articuladas, por outro lado, não são facilmente verbalizadas pelo cliente, exigindo que o especialista se aprofunde em conhecer suas experiências, interprete o que o cliente expressa e infira sobre o que ele pensa e sente. Certas necessidades Não Articuladas podem, também, ser consideradas “Necessidades Latentes” no sentido de que elas ainda não se caracterizaram como

necessidades para o cliente, mas que pode ocorrer caso estas necessidades sejam aventadas (BAYUS e SHANE, 2008).

E, por fim, também é possível fazer-se a categorização das necessidades do cliente em função de suas prioridades, visando determinar quais necessidades são “mais desejáveis” ou “úteis” frente ao conjunto de necessidades especificadas. A priorização pode ser requerida para limitar o escopo de algum projeto em razão de prazos e custos, por exemplo. De forma semelhante, pode-se considerar as necessidades usando a classificação do modelo de Kano (MIKULIĆ e PREBEŽAC, 2011) como básicas, de performance e atrativas. A básica são aquelas necessidades que não podem faltar em qualquer solução para uma dada demanda. A classificação de performance são aquelas necessidades em que quanto mais, melhor. Por fim, a atrativa são aquelas necessidades das quais o cliente não tem a expectativa, mas que, se forem atendidas, produzem uma atração ao cliente.

3.2 Conceituação de Ontologia

Ontologia tem origem na disciplina de Filosofia, mais especificamente no ramo da Metafisica, que estuda a natureza do “ser” e a “existência” (ISOTANI e BITTENCOURT, 2015). O estudo de ontologia especifica as conceituações de forma sistêmica, baseada na percepção do indivíduo, de “elementos” relevantes de uma determinada situação (ISOTANI e BITTENCOURT, 2015) (VAN HARMELEN, et al., 2008). Na área de computação, ontologias são um meio para modelar formalmente a estrutura de sistemas ou conceitos de alto nível, ou seja, das entidades e relacionamentos relevantes que emergem de sua observação e que são úteis para os propósitos dos sistemas (GUARINO, et al., 2009).

Na atividade de Requisitos de Software, frequentemente, o analista não é especialista do domínio do problema e deve aprender com o cliente sobre este domínio. Porém, entendimentos diferentes dos conceitos envolvidos do domínio, podem levar a falhas de especificação e retrabalhos (HAPPEL e SEERDORF, 2006). Assim, a utilização de ontologias tem como objetivo deixar a etapa de especificação de requisitos de software consistente e completa (ZONG-YONG, et al., 2007) (KAIYA e MOTOSHI, 2006). Um exemplo é a utilização de ontologias no mapeamento do domínio de conhecimento de negócio e no mapeamento do processamento semântico da especificação de requisitos em linguagem natural. Estes mapeamentos auxiliam o profissional responsável, principalmente, nas etapas de

elicitação de requisitos e na especificação de requisitos de software (KAIYA e MOTOSHI, 2006) (SUÁREZ-FIGUEROA e GÓMEZ-PÉREZ, 2011). As principais contribuições das ontologias estão na identificação dos problemas de negócio, na melhoria da comunicação do especialista com os *stakeholders*, na construção de modelos de documentação de software mais precisos, na rastreabilidade entre os artefatos e na melhoria da qualidade da identificação dos requisitos (ABDALAZEIM e FARID, 2021).

Contudo, percebe-se que as pesquisas sobre ontologias na atividade de Requisitos de Software estão relacionadas a domínios específicos de conhecimento de negócio (ZONG-YONG, et al., 2007) (KAIYA e MOTOSHI, 2006) (SUÁREZ-FIGUEROA e GÓMEZ-PÉREZ, 2011), não sendo aplicáveis diretamente para a atividade de Requisito de Software em quaisquer aplicações. Por esta razão, apresenta-se neste capítulo uma proposta de ontologia para a atividade de Requisitos de Software, que será empregada nesta pesquisa e que se propõe a ser genérica o suficiente para aplicação em diferentes tipos de negócio.

3.3 Proposta de Ontologia para Requisitos de Software

Como dito anteriormente, a ontologia é um estudo visando modelar formalmente os conceitos relevantes e suas relações (VAN HARMELEN, et al., 2008). No caso da atividade de Engenharia de Requisitos, este estudo tem como objetivo melhorar a qualidade da especificação, por meio de um melhor entendimento e padronização dos conceitos e relações entre os conceitos utilizados (ZONG-YONG, et al., 2007) (KAIYA e MOTOSHI, 2006).

A ontologia proposta nesta dissertação reflete os conceitos estabelecidos na PBSRS, abordagem na qual esta pesquisa de mestrado se fundamenta. A abordagem PBSRS possui conceitos e um mapeamento entre domínios (*i.e.*, Problemas, Necessidades e Requisitos) para realizar a atividade de especificação de requisitos de forma mais sistemática, estabelecendo as relações causais existentes entre os domínios (SOUZA e STADZISZ, 2016). A partir dos conceitos da PBSRS, a ontologia proposta trouxe uma maior precisão nas definições dos conceitos e relações e incluiu contribuições de novos conceitos ligados aos *stakeholders* e suas relações.

A Figura 9 apresenta o modelo proposto que, junto com o tesauro (ver Apêndice A), constituem a ontologia desenvolvida. Este modelo, denominado

“esquema” (*i.e.*, *schema* em inglês) foi construído utilizando a linguagem *Ontology Web Language* (ZONG-YOUNG, et al, 2007). Refere-se a uma representação formal e explícita de conceitos, classes e relações entre entidades dentro de um domínio específico.

Os retângulos representando os conceitos na Figura 9 possuem cores usadas para indicar grupos de conceitos correlatos. Este artificio não faz parte da ontologia, sendo apenas um recurso para facilitar sua interpretação. A cor vermelha indica os conceitos relacionados com o *Stakeholder*. A cor roxa foi empregada para indicar os conceitos relacionados com o Domínio dos Problemas. A cor verde foi usada para indicar os conceitos relacionados com o Domínio das Necessidades e, por fim, a cor azul foi utilizada para indicar os conceitos relacionados com o Domínio dos Requisitos. Adicionalmente, alguns conceitos têm mais de uma cor, o que indica que eles estão relacionados com mais de um domínio.

Observa-se, na Figura 9, que esta ontologia é constituída por um conjunto de conceitos (desenhados na forma de retângulos com bordas arredondadas) e relações (desenhadas na forma de arcos dirigidos entre entidades). De acordo com Isotani e Bittencourt (2015), um conjunto de conceitos de um determinado domínio de negócio envolve conceitos essenciais resultantes da articulação do conhecimento básico deste domínio e pode ser representando por meio de um vocabulário especializado. Este corpo de conhecimento básico é organizado em uma hierarquia de classes, resultante de relações “*is-a*” (*i.e.*, é-um ou é-uma) entre conceitos (ISOTANI e BITTENCOURT, 2015).

Observa-se que, em uma ontologia, cada conceito pode ser entendido como uma classe de conceito, no sentido de representar um conjunto de instâncias daquele conceito. Por exemplo, o conceito “*Stakeholder* Externo” representa qualquer pessoa ou instituição neste papel em relação ao software, como um consultor, um investidor ou uma empresa parceira. Assim, “*Stakeholder* Externo” é uma classe e “um consultor” é uma instância desta classe.

Adicionalmente, outras relações importantes entre os conceitos, como “*part of*” (*i.e.*, parte de) e “*domain*” (*i.e.*, é uma propriedade de) podem, também, ser estabelecidas, além de eventuais restrições semânticas (ISOTANI e BITTENCOURT, 2015). Por exemplo, a relação “tem interesse”, que conecta o conceito “*Stakeholder*” ao conceito “Interesse”, indica que todo *stakeholder* possui um interesse com relação ao software.

Figura 9 - Esquema da proposta de ontologia para Requisitos de Software

Fonte: Autoria própria (2024)

No esquema da Figura 9, algumas relações entre os conceitos são baseadas em *Resource Description Framework* (RDF), mais especificamente RDF

Schema ou RDF-S. RDF é um *framework* de especificações feito pela Consócio *World Wide Web Consortium* (W3C), planejado como um modelo de dados para metadados. Equivale a uma linguagem de representação de informação na *Web*, permitindo que os recursos possam ser descritivos formalmente e sejam acessíveis por máquinas (ISOTANI e BITTENCOURT, 2015). Um RDF *Schema* é uma extensão semântica de RDF que descreve um conjunto de recursos e relacionamentos entre eles.

No esquema da Figura 9, por exemplo, apresenta-se **Cliente "rdsf: *SubClassOf*" *Stakeholder*** que significa que a classe Cliente "é uma subclasse" da classe *Stakeholder*. Outro exemplo é **Nome "rdsf: *domain*" *Stakeholder*** que significa que Nome é uma propriedade da classe *Stakeholder*.

Na seção a seguir, os conceitos de *Stakeholders*, Problema, Necessidade e Requisito serão detalhados.

3.4 Conceituação de Stakeholder

Todo produto, incluindo software, é desenvolvido para atender a demanda de algum mercado alvo. Diferentes atores neste mercado alvo podem ter participação no uso, seleção e aquisição, entre outros, do software. Estes atores recebem diferentes denominações na literatura e no cotidiano das empresas, como "cliente", "consumidor", "usuário" e "público" (SUH, 1990) (GROEN, et al., 2017). Emprega-se, também, o termo "*stakeholder*", neste contexto, para se fazer referência genérica a qualquer ator do mercado alvo ou interno da empresa que tenha algum "interesse" no produto, em outras palavras, que tenha alguma responsabilidade, participação ou influência no produto e na sua aquisição (BOURQUE e FAIRLEY, 2014) (INCOSE, 2015).

Assim, entende-se que um "*Stakeholder*" é um conceito que remete àquele (*i.e.*, um indivíduo ou coletivo de indivíduos) ou àquilo (*i.e.*, uma organização) que possui um ou mais interesses associados ao software a ser desenvolvido. Este "Interesse" do *stakeholder* está relacionado com a sua percepção de relevância, importância, vantagem ou utilidade frente ao "Objeto de Interesse", que remete a algo que considera útil (KROLL, 1991) (SILVIA, 2008) (SINAGA, et al., 2021). Cada "Interesse" poderia variar em "Importância" conforme o julgamento do *stakeholder* (SILVIA, 2008), levando em conta a potencial utilidade ou contribuições do produto aos seus negócios. Entretanto, trata-se de uma avaliação preliminar e superficial. Na

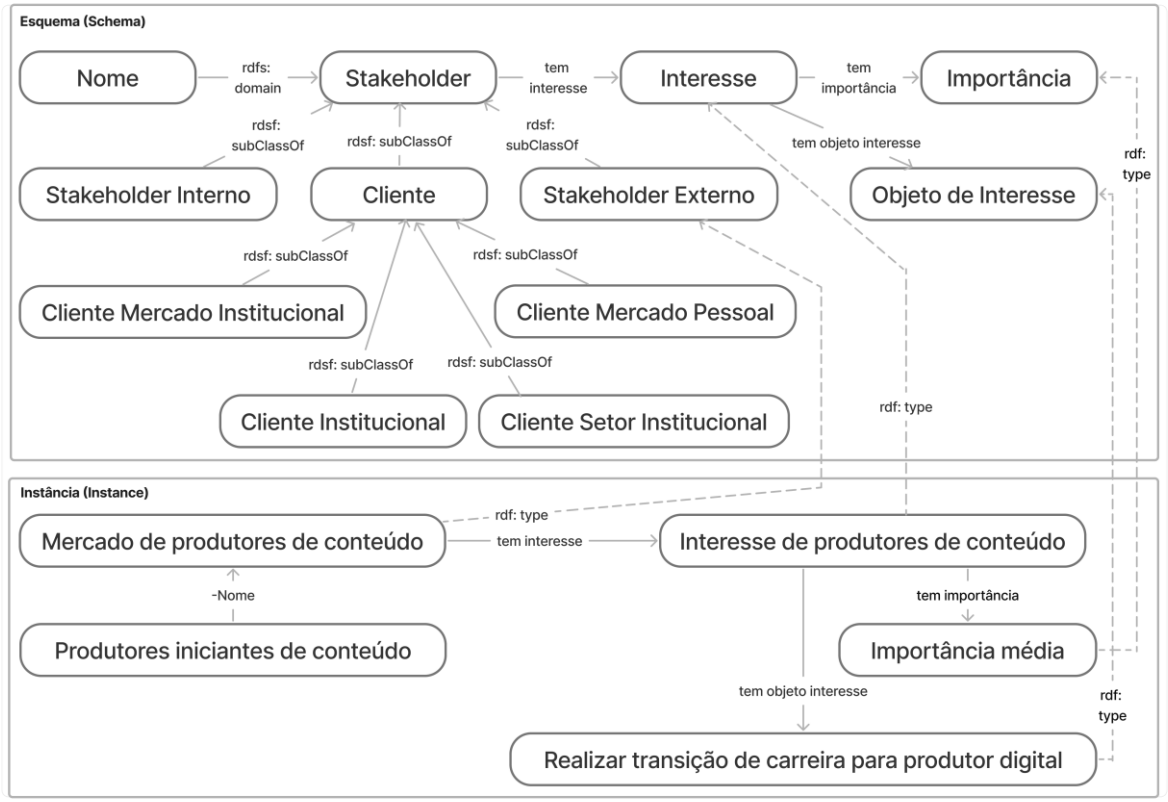
abordagem PBSRS, entende-se que quando um *stakeholder* tem “interesses” por um determinado produto ou proposta de solução (*i.e.*, produto que viria a ser desenvolvido ainda) é porque ele possui “problemas” relacionados que serão determinados no Domínio do Problema (SOUZA e STADZISZ, 2016).

De acordo com o produto de software e seus mercados, os *“Stakeholders”* envolvidos podem ser múltiplos e variados. Nesta pesquisa, propôs-se uma classificação dos *stakeholders* em: “Cliente”, “*Stakeholder* Interno” e “*Stakeholder* Externo”. O “Cliente” foi definido como aquele ou aquilo que irá adquirir e utilizar o produto a ser desenvolvido, podendo ser um consumidor, um comprador, um adquiridor, um usuário, dentre outros. As demais classes de *“Stakeholders”* (*i.e.*, “*Stakeholder* Interno” e “*Stakeholder* Externo”) têm semelhança em exercer alguma influência sobre o produto e negócio relacionado, contudo não expressam compradores, nem usuários do produto. O “*Stakeholder* Externo” é aquele ou aquilo que não está inserido na rotina da empresa proponente do software, como fornecedores, governos, Organizações Não Governamentais e associações. Por outro lado, o “*Stakeholder* Interno” é aquele ou aquilo que se encontra dentro do ambiente da empresa proponente do software (*e.g.*, acionistas e colaboradores da organização, setores da empresa, responsáveis, diretores e desenvolvedores).

A subclasse “Cliente”, por sua vez, possui uma classificação em tipos (ou subclasses) de clientes que adquirem e/ou usam o produto, podendo ser um “Cliente Mercado Institucional”, “Cliente Institucional”, “Cliente Setor Institucional” e “Cliente Mercado Pessoal”. Um exemplo de “Cliente Mercado Institucional” é o “Mercado de Instituições Financeiras”, ou melhor dizendo, o produto que será desenvolvido tem como público-alvo o “Mercado de Instituições Financeiras”. Outra subclasse é o “Cliente Institucional” que é o público de uma instituição específica como, por exemplo, uma indústria de laticínios em particular. Já a subclasse “Cliente Setor Institucional” compreende o público de um determinado setor dentro de uma instituição específica, que irá adquirir um produto que atenda particularmente suas necessidades, como, o diretor e os funcionários do setor de estoque de uma fábrica de automóveis. Por fim, a subclasse “Cliente Mercado Pessoal” representa uma parte do mercado de pessoas físicas que poderá adquirir e usar o produto. Um exemplo desta subclasse são os “Estudantes Universitários” que seriam entendidos como potenciais compradores e usuários de um certo produto.

A Figura 10 representa a parte do esquema envolvendo o conceito de *Stakeholders* e conceitos relacionados.

Figura 10 – Esquema de conceituação de *Stakeholder*



Fonte: Autoria própria (2024)

O exemplo apresentado na parte inferior (*i.e.*, Instância) da Figura 10 é o caso de um "*Stakeholder*" da classe "*Stakeholder* Externo", denominado "Mercado de produtores de conteúdo" que tem o nome "Produtores iniciantes de conteúdo". Seu "Objeto de Interesse" é "Realizar a transição de carreira para produtor digital". Uma vez que, muitos produtores iniciantes de conteúdos, normalmente, mantêm o seu emprego atual nesta transição, considera-se que a "Importância" deste "Interesse" é "Importância média", pelo fato de o produtor iniciante já ter uma fonte de renda para manter as suas despesas e possuir uma dupla jornada de trabalho.

3.5 Conceituação de Domínio dos Problemas

De acordo com Souza e Stadzisz (2016), os problemas representam o conhecimento a respeito do que motiva ou leva um *stakeholder* a adquirir algum

produto. Na conceituação de *stakeholder*, o conceito de “Interesse” expressa, de forma bastante abstrata, a motivação original de um *stakeholder* por uma solução ou produto. Esse interesse leva à busca pelo entendimento de qual é o problema ou quais são os problemas que fundamentam esse interesse. Em outras palavras, os problemas que um *stakeholder* sofre caracterizam seu interesse. Pode-se dizer, também, que um *stakeholder* que não tem ou não percebe problemas sobre alguma questão ou negócio no qual tem envolvimento, não tem interesse em alguma solução a respeito. Por exemplo, quando um *stakeholder* tem um problema em “pagar os salários de seus funcionários dentro do prazo”, ele poderia, imaginar um sistema de gestão de RH para garantir o pagamento dos salários dos funcionários dentro do prazo. Esta especificação abstrata da solução visando resolver um problema, na abordagem PBSRS, é denominada “*Glance*”.

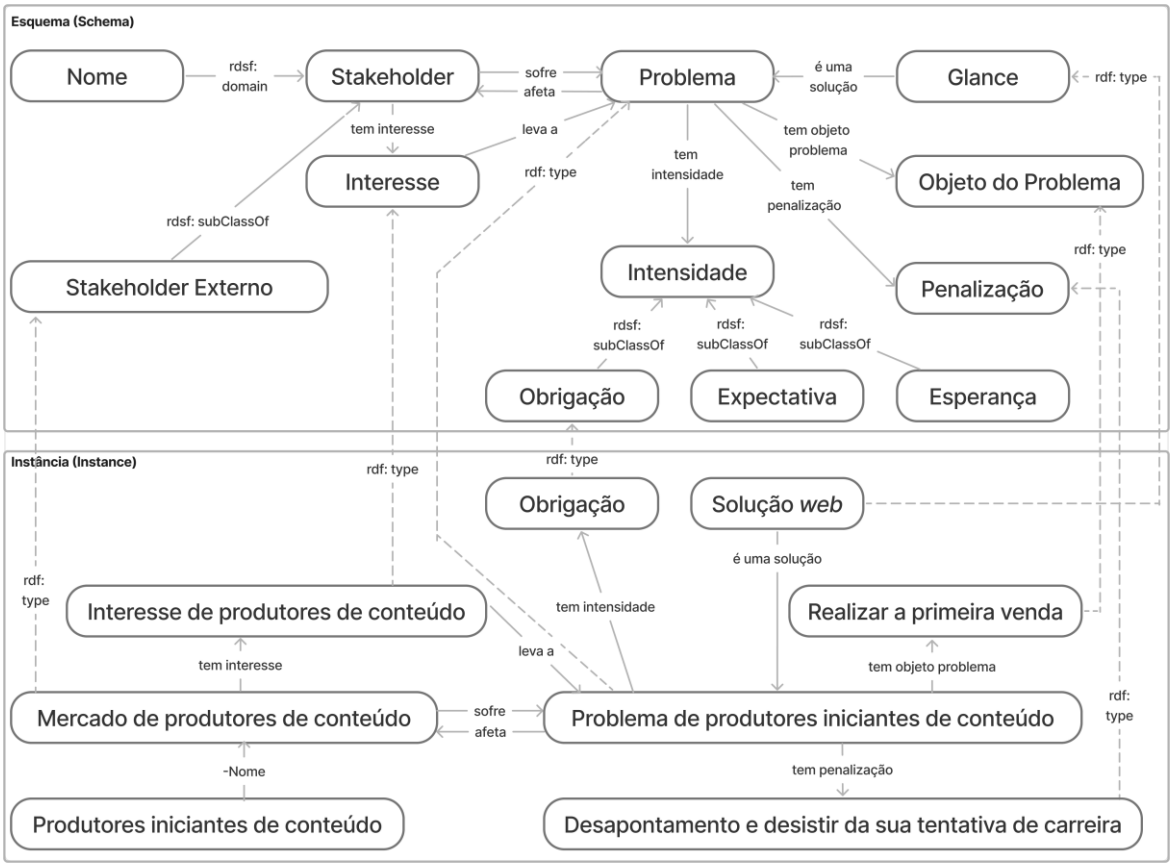
Considera-se que o conceito de “Problema” do *stakeholder* é definido como “o que pode causar ou está causando um efeito negativo no *stakeholder* em relação ao seu(s) negócio(s)”, que pode ser percebido por meio de métricas e indicadores (*i.e.*, por meio de observação da condução dos negócios, do relacionamento com o mercado, do atendimento às regulamentações, entre outros) e de maneira menos explícita na forma de sentimentos dos *stakeholders* a respeito dos negócios, como riscos, tendências, pressões de mercado e visões estratégicas” (SOUZA e STADZISZ, 2016).

A interpretação para o conceito de “Problema”, proposta na PBSRS e adotada nesta pesquisa, é original, entendendo que um problema é uma “obrigação que recai sobre o *stakeholder*”. Neste sentido, decorre que todo problema deve ter uma “punição” que caracteriza a obrigação. Esse aspecto, inclusive, é fonte de mal entendimento frequente na análise de problemas. Considere, como exemplo, a situação de um indivíduo que, após algum tempo sem se alimentar, sente fome. Fome é, certamente, uma dor, que pode, até, assumir uma intensidade muito grande. Não seria incomum ouvir que esta “fome” seria um problema, porque é uma dor sentida por alguém. Neste sentido, nas empresas fala-se comumente sobre as “dores dos clientes”. A questão é que o sentimento de fome não é o problema e sim um alerta do organismo, ou seja, é a consequência de algo. O problema (*i.e.*, a obrigação) é, na verdade, o dever do indivíduo de se nutrir, que quando não cumprido leva à penalização de fome.

Para detalhar o conceito de Problema, definiu-se que ele se compõe de três outros conceitos, que são: “Objeto do Problema”, “Penalização” e “Intensidade”. O conceito de “Objeto do Problema” remete a algo que descreve a dificuldade que é a fonte do problema. Dito de outro forma, o “Objeto do Problema” descreve a obrigação à qual o *stakeholder* está submetido. Cada problema é, também, caracterizado por uma “Penalização” que representa a consequência do não cumprimento da obrigação (*e.g.*, sinalização, dor, prejuízo). Ao identificar a penalização, pode-se perceber qual é a intensidade do problema. Na PBSRS, a especificação de intensidade possui três subclasses: Obrigação, Expectativa e Esperança (SOUZA e STADZISZ, 2016). A “Obrigação” é uma classe de problemas que determina um dever do *stakeholder* devido à imposição de uma penalidade (SOUZA e STADZISZ, 2016). Um exemplo seria a obrigatoriedade de declaração anual de ajuste de imposto de renda, que tem como penalização uma multa em caso de não cumprimento. A classe “Expectativa” indica que o *stakeholder* sofre uma expectativa oriunda de seus próprios clientes (*i.e.*, *stakeholders*) e que o não atendimento desta expectativa, mesmo que não seja obrigatória em um sentido estrito, produz uma penalização. Entende-se por expectativa que alguém ou alguma coisa (*e.g.*, empresa ou mercado) espera algo do *stakeholder*, no sentido de considerar que esse algo é factível, possível e desejado, mesmo que não seja obrigatório. Por exemplo, os clientes de um fabricante de automóveis podem ter a expectativa de que novos modelos serão melhores em termos de redução de emissões, além das exigências regulatórias, caso contrário, poderá haver uma redução de atratividade sobre os automóveis ofertados (SOUZA, 2016). Por fim, a classe “Esperança” representa, também, algo que alguém ou alguma coisa espera do *stakeholder*, porém de maneira menos incisiva ou mais branda do que as obrigações e expectativas, mas, ainda assim, podendo produzir uma penalização em caso de não atendimento. Uma esperança é algo que se imagina ser possível e que o *stakeholder* poderia fazer ou prover e que poderia ser entendido como um diferencial ou alguma coisa a mais que é ofertada. Por exemplo, os clientes de uma empresa de telecomunicações têm a esperança de receber uma comunicação de felicitações pelo seu aniversário ou o cliente de uma oficina mecânica receber uma lavagem gratuita do carro após a revisão.

Isto posto, o esquema da Figura 11 apresenta os conceitos e relacionamentos do domínio do problema e ilustra um exemplo de instância destes conceitos.

Figura 11 – Esquema de conceituação de Domínio dos Problemas



Fonte: Autoria própria (2024)

No exemplo ilustrado na Figura 11, a instância apresenta um exemplo de especificação de problema, associando-o aos conceitos da ontologia proposta. Neste exemplo, o *stakeholder*, identificado por “Mercado de produtores de conteúdo”, é uma instância da classe “*Stakeholder* Externo”, que é uma subclasse de *Stakeholder*. Este *stakeholder* possui o nome “Produtores iniciantes de conteúdo” e tem um interesse identificado por “Interesse dos produtores iniciantes”, qualificado pelo objeto de interesse “Tornar-se um produtor digital”, considerado por este *stakeholder* como de “Importância média”. Este interesse leva à identificação do problema (*i.e.*, um dos possíveis problemas) que afeta este *stakeholder*, identificado por “Problema de produtores iniciantes de conteúdo”. Este problema tem como

objeto “Realizar a primeira venda”, cuja intensidade é do tipo “Obrigação”. Caso o produtor iniciante de conteúdo não realize a primeira venda ou haja uma demora considerável desta venda, a sua “Penalização” é de “Desapontamento e desistir da sua tentativa de carreira”. Textualmente, o problema relatado pode ser expresso da seguinte forma:

Produtores iniciantes de conteúdo têm obrigação de realizar a primeira venda sob pena de desapontamento e desistência de sua tentativa de carreira como produtores de conteúdo digital.

A partir da identificação dos *stakeholders* e de seus problemas, são concebidas e especificadas as Necessidades do *Stakeholder*, conforme discutido na seção a seguir.

3.6 Conceituação de Domínio das Necessidades

Como dito anteriormente, um “problema” é algo que causa ou pode causar um efeito negativo ao *stakeholder* e que, por consequência, cria uma motivação em buscar por um produto, podendo ser um software, para prover a resolução ou minimização deste problema e de seus efeitos (*i.e.*, punições). A partir do conhecimento dos problemas dos *stakeholders* (*i.e.*, Domínio dos Problemas), pode-se idealizar e especificar as necessidades do *stakeholder* que descrevem o que o futuro produto “entregará” de útil para os *stakeholders*, compondo o Domínio das Necessidades. Pertinente lembrar que este domínio não define o que o software fará (*i.e.*, requisitos funcionais), tampouco em quais situações será utilizado (*i.e.*, casos de uso), nem quais são suas características (*i.e.*, requisitos não funcionais).

A carência ou falta expressa por uma necessidade é fruto de uma causa anterior, que, como indicado na PBSRS, corresponde aos problemas. Assim, toda percepção de carência ou falta ocorre pela existência de algum problema motivador. Na verdade, cada necessidade expressa algo do qual se carece em vista da resolução ou minimização de algum problema, junto com a idealização (*i.e.*, *Glance*) da resolução deste problema. Considere-se o problema da obrigação de nos nutrirmos regularmente, cuja punição é a dor da fome e o risco de subnutrição e de sobrevivência. A partir deste enunciado, pode-se perguntar o que é “necessário” ou o que falta ao indivíduo para minimizar ou resolver este problema. Uma resposta

seria a de que é necessário que o indivíduo se nutra adequadamente, o que corresponde a uma necessidade vinculada àquele problema. Entende-se, então, que as necessidades são expressões de soluções para os problemas dos *stakeholders*. Por exemplo, “nutrir-se” (*i.e.*, necessidade) é uma solução para “ser obrigado a estar nutrido” (*i.e.*, problema).

Na PBSRS, vai-se um pouco além na definição das necessidades. Para que não haja confusão entre os conceitos de necessidades e requisitos e para que cada um destes dois conceitos represente uma fase distinta e encadeada do processo de especificação de software, entende-se que as “Necessidades” devem se limitar a especificar “o que se precisa” do ponto de vista do que o software deverá “entregar” ou “prover” e não do que ele deverá fazer (que seria a expressão dos requisitos). Considere, como exemplo, o problema de um fazendeiro em que “ele deve proteger a plantação em caso de geada, sob pena de dano e perda da lavoura.”. Podem decorrer deste enunciado diversas ideias de solução (*i.e.*, *Glance*), inclusive a de ligar aspersores de água sobre a plantação quando a temperatura ambiente estiver abaixo de um certo patamar. Neste sentido, uma das coisas que o fazendeiro necessita (*i.e.*, que ele não tem) é ter conhecimento de quando agir, ligando os aspersores. A expressão “necessita ter conhecimento de” não diz o que o futuro software faria (que seria um requisito funcional), mas sim o que ele entregaria (*i.e.*, um conhecimento).

Por fim, entende-se, ainda, que quando se especifica uma necessidade, pode-se ou deve-se identificar a “Fonte” (*e.g.*, meio, sistema ou produto) que proveria o que o *stakeholder* tem carência, ou seja, a coisa que se quer que seja provida (*i.e.*, “Objeto da Necessidade”). Embora, até certo ponto, o *stakeholder* possa não saber o que proveria suas carências ou possa existir mais de uma possibilidade de meios para prover suas carências, a identificação da “Fonte” para provimento das carências do *stakeholder* é uma informação importante na alocação deste provimento. Esta identificação permite estabelecer claramente qual solução terá a responsabilidade de prover cada necessidade. Considerando o exemplo anterior, pode-se dizer que “o fazendeiro necessita de um software (*i.e.*, fonte) para ter conhecimento da temperatura ambiente”. Desta forma, a necessidade torna-se orientada a uma solução, que neste exemplo seria a aspersão.

Quando a “Fonte” de uma necessidade for um software, pode-se determinar o “Tipo da Necessidade” de acordo com as seguintes subclasses: criação de objetos

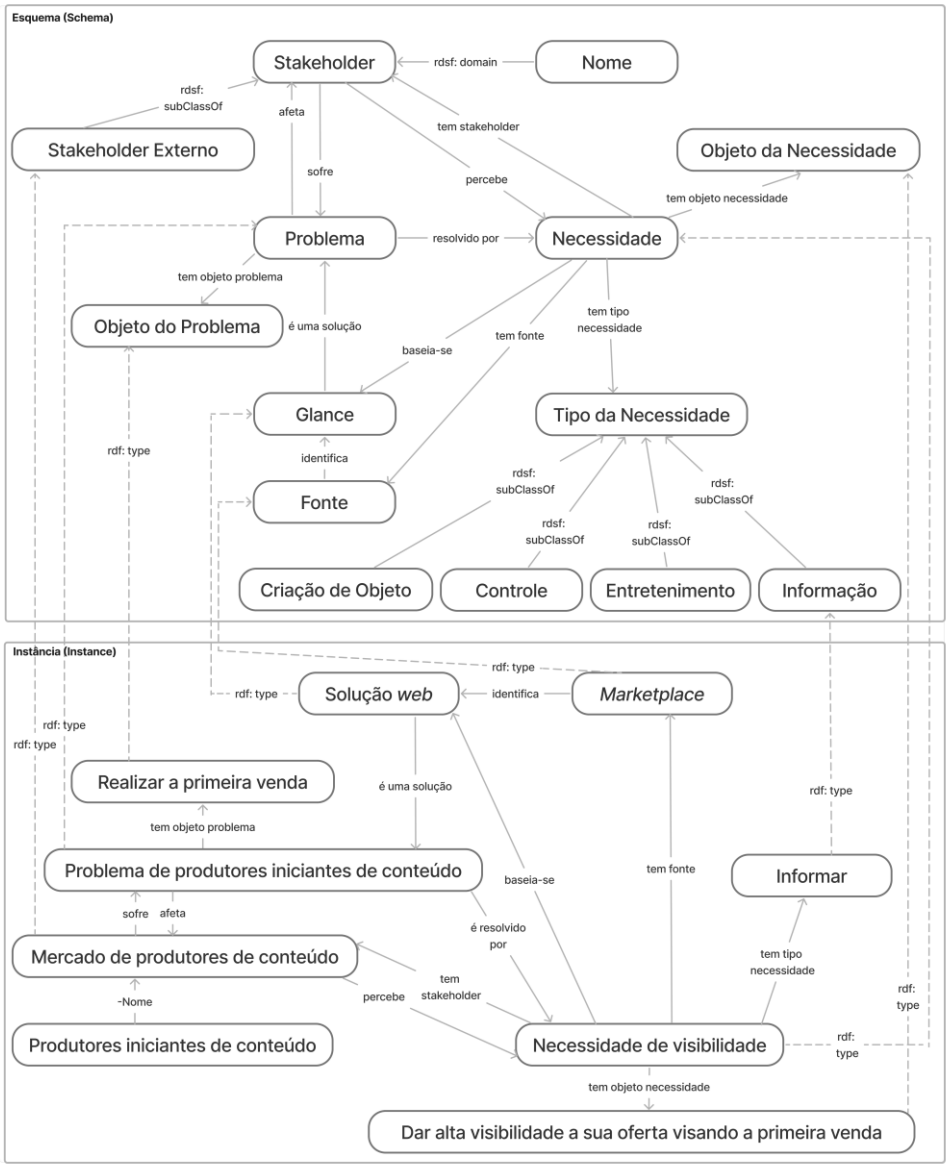
digitais, controle, entretenimento e informação (SOUZA e STADZISZ, 2016). Estas subclasses representam os principais grupos de coisas que um software pode prover e podem ajudar no processo de identificação das necessidades associadas à resolução dos problemas dos *stakeholders*. A “criação de objetos digitais” representa a capacidade dos softwares em prover meios para criar objetos digitais na forma de textos, gráficos, modelos, tabelas, apresentações, fórmulas, vídeos, entre outros (SOUZA e STADZISZ, 2016). O “controle” ou “automação” é, também, algo que os softwares podem prover envolvendo o monitoramento e ação contínua seja sobre dados (*e.g.*, um sistema de informação) ou sobre ambientes físicos (*e.g.*, um sistema de climatização, um sistema de segurança ou um sistema de produção) (SOUZA e STADZISZ, 2016). Como exemplo, um software poderia prover o automatismo de abertura e fechamento de uma porta automática de aeroporto. Softwares podem, ainda, fornecer diversão, distração, lazer, recreação ou passatempos, classificados como “necessidades de entretenimento”. Exemplos seriam softwares para jogos, reprodutores de música e vídeo, leitores de textos e mídias sociais (SOUZA e STADZISZ, 2016). Por fim, umas das principais contribuições dos softwares é, seguramente, prover “informação” que auxilia na construção de conhecimento para os usuários (SOUZA e STADZISZ, 2016). Para este tipo de necessidade, há inúmeros softwares, incluindo aqueles de “sistemas de informação”, *websites* e aplicações *web*, que provêm informações a seus usuários na forma de notícias, relatórios, *dashboards*, estatísticas, tabelas de dados e textos (SOUZA e STADZISZ, 2016). Pode-se, assim, dizer que um cliente poderia “necessitar” de um software para ter conhecimento ou saber de algo, como os resultados dos jogos de um campeonato ou a previsão do tempo para o dia seguinte.

A Figura 12 apresenta um fragmento do esquema da ontologia proposta, destacando o esquema do Domínio das Necessidades e, abaixo, uma instância de exemplo deste domínio. O exemplo envolve um *stakeholder* identificado por “Mercado de produtores de conteúdo” da classe “*Stakeholder* Externo”, cujo nome é "Produtores iniciantes de conteúdo”. Este *stakeholder* sofre um problema identificado por “Problema de produtores iniciantes de conteúdo”, cujo objeto é “Realizar a primeira venda”. Uma alternativa de solução parcial para este problema é identificada pela necessidade “Necessidade de Divulgação”, classificada como “Informação” e que tem como objeto “Dar alta visibilidade a sua oferta visando a primeira venda”. Esta necessidade seria provida por um software *web* de venda de

conteúdo digital, denominado “*Marketplace*”. Textualmente, a necessidade relatada pode ser expressa de acordo com a formatação da especificação de necessidade sugerida na abordagem PBSRS (SOUZA e STADZISZ, 2016):

Produtores iniciantes de conteúdo precisam de um Marketplace para dar alta visibilidade a sua oferta visando a primeira venda.

Figura 12 – Esquema de conceituação de Domínio das Necessidades



Fonte: Autoria própria (2024)

Após a identificação das necessidades do *stakeholder*, realiza-se a especificação de requisitos de software, cujo conceitos são apresentados na seção seguinte.

3.7 Conceituação de Domínio dos Requisitos

O Domínio de Requisitos contém o conhecimento sobre o que o software deverá fazer (*i.e.*, requisitos funcionais) e ter como características (*i.e.*, requisitos não funcionais). Na abordagem PBSRS, os requisitos decorrem do que foi especificado no Domínio das Necessidades, devido a sua dependência causal com as necessidades que foram identificadas (SOUZA e STADZISZ, 2016). Em outras palavras, a especificação do que o produto (*i.e.*, software) poderá entregar para o cliente (*i.e.*, necessidades) é a causa da identificação do que o software deverá fazer ou ter como características.

O conceito de “Requisito” remete a algo que é exigido pelo *stakeholder*, no sentido do que o futuro software deverá fazer ou ter como característica para que uma ou mais necessidades sejam atendidas. Na ontologia proposta, o conceito de “Requisito” é desdobrado nas subclasses “Funcional” e “Não Funcional”, como é classicamente feito na literatura (BOURQUE e FAIRLEY, 2014). Este conceito está relacionado com o de Necessidade, uma vez que cada requisito decorre de uma ou mais necessidades, e está relacionado com o conceito de *Vision*, dado que cada requisito representa uma exigência técnica relacionada com a solução. O conceito de “*Vision*” representa um detalhamento da solução visando definir o seu escopo, ou seja, uma descrição mais detalhada do conceito de “*Glance*”, mas que ainda não é o detalhamento do projeto da solução, propriamente dito. Assim, todos os requisitos devem se basear na *Vision*, de forma que todos sejam compatíveis com a solução delineada.

O conceito de “Requisito funcional” é detalhado por meio dos conceitos de “Sujeito”, “Verbo de Ação”, “Objeto do Requisito”, “Restrições” e “Condições”, de acordo com a PBSRS (SOUZA e STADZISZ, 2016). O conceito de “Sujeito” designa a solução ou uma parte ou componentes desta que é alvo da exigência funcional expressa pelo requisito. Por exemplo, na especificação de requisitos de um software, o “software” ou o “nome do software” ou, ainda, “um certo componente do software” seriam os sujeitos do requisito. O conceito de “Verbo de Ação” caracteriza a exigência imposta pelo requisito e, também, a ação ou função exigida do sujeito do

requisito. Comumente, a exigência é expressa pelo verbo “dever”, conjugado no futuro (*i.e.*, deverá ou *shall* quando em inglês). O verbo de ação, em geral, é um verbo conjugado no infinitivo para remeter à ideia de ação efetivamente. Exemplos de expressão seriam: “O software deverá registrar ...”, “O SGP deverá gerar ...” e “O robô Laura deverá identificar ...”. O conceito de “Objeto do requisito” estabelece o que sofre a ação do “Verbo de Ação”, ou seja, o objeto da ação. Exemplos seriam: “O software deverá registrar a ocorrência de um alarme.”, “O SGP deverá gerar uma notificação ao usuário.” e “O robô Laura deverá identificar uma infecção.”. O conceito de “Restrições” representa, quando houver, propriedades ou características, em geral limitantes, à ação prevista no requisito. Um exemplo de restrição (temporal) seria: “O software deverá ativar o alarme de incêndio em no máximo 2 segundos.”. Deve-se notar que o conceito de “Restrições” representa um elemento não funcional associado ao objeto do requisito. Assim, alternativamente, as restrições poderiam ser especificadas separadamente na forma de requisitos não funcionais e fazendo referência ao requisito funcional associado. Por fim, um requisito pode conter, opcionalmente, a informação de quando se aplica, na forma do conceito de “Condições”. Um exemplo seria: “O software deverá ativar o alarme de incêndio em no máximo 2 segundos, quando receber o sinal do sensor.”.

Com relação ao conceito de “Sujeito”, propõe-se empregar três subclasses: “Fonte”, “Parte da Fonte” e “Funcionalidade”. O conceito de “Fonte” é o mesmo que o empregado na especificação de necessidades, que determina a fonte, ou seja, o que provê o que é necessário. Este conceito equivale ao sujeito de requisitos correspondendo ao que realizará as ações que são requisitadas em vista de prover o que é necessário aos *stakeholders*. Um exemplo seria a necessidade “O cliente precisa de um software *web* para ter conhecimento do resultado de vendas do mês.” da qual decorrem os requisitos “O software *web* deverá permitir registrar uma venda realizada.” e “O software *web* deverá permitir consultar o resultado das vendas do mês.”. Neste exemplo, o sujeito dos requisitos corresponde à fonte da necessidade. O conceito de “Parte da Fonte” permite indicar que um requisito se aplica, especificamente, a um componente da solução e não à totalidade da solução. Um exemplo seria: “O módulo de *login* deverá checar o *id* e senha digitados.”. Neste caso, o “módulo de *login*” constituiu um dos componentes de um software maior. Por fim, o conceito de “Funcionalidade” se aplica, unicamente, a sujeitos de requisitos não funcionais quando a expressão de restrições ou características se associa a

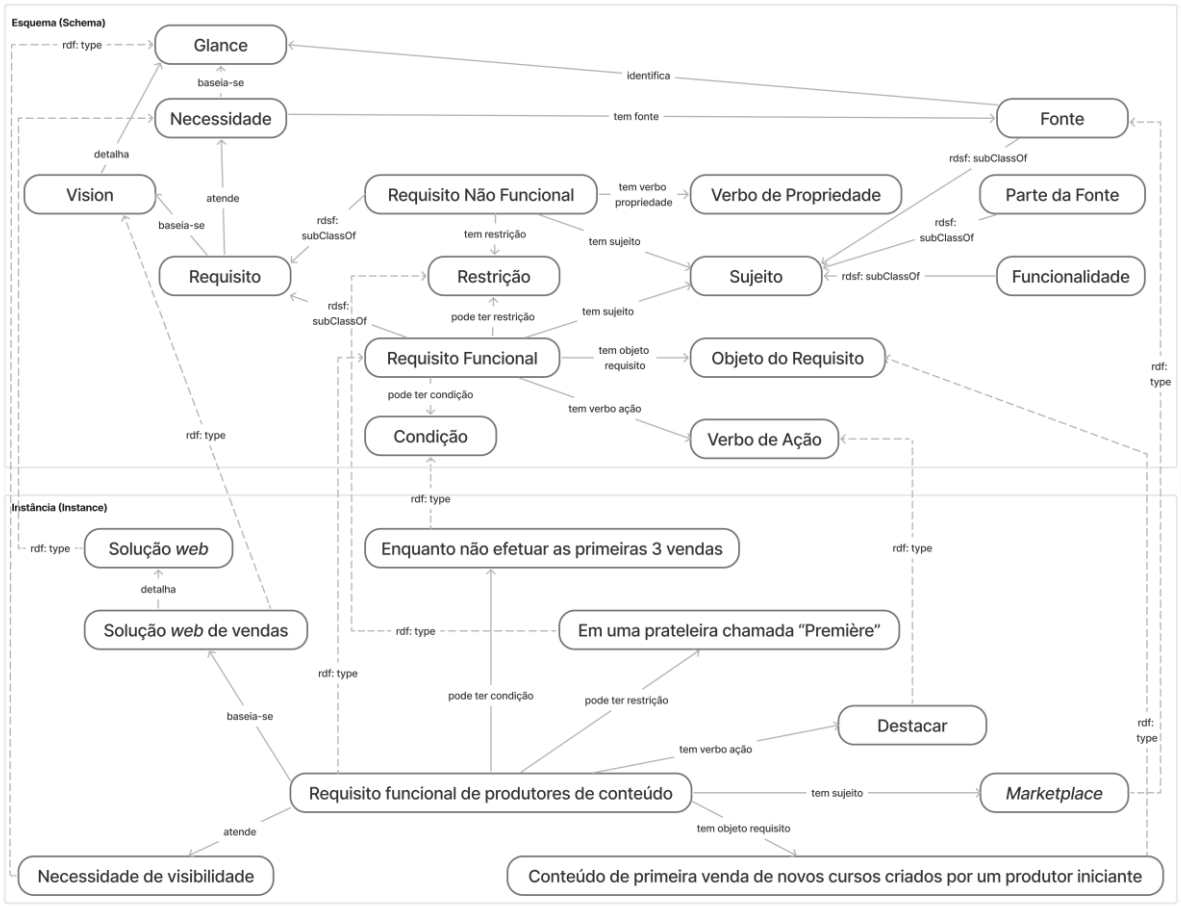
execução de uma ação expressa por um requisito funcional previamente especificado. Um exemplo seria: “A geração de alarme deverá ser feita em até 2 segundos.”. Neste caso, a “geração de alarme” é uma ação expressa em um requisito funcional do tipo “O software deverá ativar o alarme de incêndio" e, por isso, este sujeito é classificado como Funcionalidade.

O conceito de “Requisito Não Funcional” é utilizado na especificação de requisitos que expressam restrições, características e propriedades, em vez de exigências de ações ou funções feitas por meio de requisitos funcionais. Este conceito se detalha nos conceitos de “Sujeito”, “Verbo de Propriedade” e “Restrições”. O conceito de Sujeito é análogo ao empregado nos requisitos funcionais. O conceito de “Verbo de Propriedade” remete à existência de restrições expressas pelo requisito. Mais comumente, empregam-se os verbos “ser”, “ter” e “conter” neste contexto. Como os requisitos não funcionais remetem a exigências que não são funcionais, eles podem incluir uma diversidade de outras restrições que poderia, ainda, ser categorizadas como econômicas, estéticas, de desempenho, operacionais e ambientais. Exemplos seriam: “O software deverá ser desenvolvido em linguagem C++.” e “O software deverá ter IHC em tons de cinza.”.

A Figura 13 ilustra o fragmento de esquema da ontologia proposta envolvendo o Domínio dos Requisitos de acordo com o que foi apresentado. A figura apresenta, também, uma instância do esquema por meio de um exemplo. Neste exemplo, a “Solução *web*”, que representa o *Glance*, é detalhada na “Solução *web* de vendas” que representa sua “*Vision*”, na qual baseia-se a necessidade “Necessidade de produtores de conteúdo”. O requisito “Requisito funcional de produtores de conteúdo” atende a necessidade e tem como sujeito o “*Marketplace*” que é a “Fonte” da necessidade. O “Verbo de Ação” deste requisito é “Destacar” e o “Objeto do Requisito” é “Conteúdo de primeira venda de novos cursos criados por um produtor iniciante”. Como “Restrições” foi indicado que o destaque será “Em uma prateleira chamada Première”, tendo como “Condição” “Enquanto não efetuar as primeiras 3 vendas” para aplicação do requisito. Textualmente, o requisito pode ser expresso da seguinte forma:

O Marketplace deverá destacar o conteúdo de primeira venda de novos cursos criados por um produtor iniciante em uma prateleira chamada “*Première*”, enquanto não efetuar as primeiras 3 vendas.

Figura 13 – Esquema de conceituação de Domínio dos Requisitos



Fonte: Autoria própria (2024)

3.8 Discussão sobre o Capítulo

Este capítulo descreveu o conceito de "necessidades dos clientes" e apresentou uma proposta de ontologia para a especificação de requisitos de software envolvendo os conceitos e relações associados aos *stakeholders* e domínios dos problemas, das necessidades e dos requisitos. Os conceitos e relações contidos nesta ontologia cobrem aqueles propostos na PBSRS. Este estudo apresenta, também, alguns conceitos e relações adicionais propostos nesta dissertação e que complementam os da PBSRS. Assim, por meio da concepção da ontologia da PBSRS, este estudo apresentou as seguintes contribuições:

- Aprofundamento do conceito de Necessidades do Cliente;

- Definição dos conceitos e relações associados aos *stakeholders* incluindo: conceitos de *Stakeholder*, Cliente, *Stakeholder* Externo e *Stakeholder* Interno, subclassificação de Cliente, Interesse e relacionamentos associados;
- Detalhamento de conceitos e relações associados ao Domínio dos Problemas, incluindo o conceito de *Glance*, relacionamento de Problema com Interesse e *Glance*;
- Detalhamento de conceitos e relações associados ao Domínio das Necessidades, incluindo o conceito de Tipo de Necessidade e relacionamento de Necessidade com *Glance*; e
- Detalhamento de conceitos e relações associados ao Domínio dos Requisitos (*i.e.*, o conceito e relacionamentos com *Vision*).

O objetivo da proposição desta ontologia foi o de estabelecer, de maneira mais precisa, todos os conceitos envolvidos na especificação de requisitos segundo a abordagem PBSRS, de forma que o processo de Requisitos de Software possa seguir uma sistemática compreensível e menos sujeita a variações de interpretações dos conceitos utilizados. Entende-se que, assim, pode-se reduzir os riscos de má formulação dos requisitos que podem implicar consequentes desvios do produto final frente aos interesses dos clientes.

4 DESCRIÇÃO DA TÉCNICA PROPOSTA

Este capítulo apresenta a proposta de pesquisa, incluindo a estruturação da associação entre os domínios. Também será apresentado uma visão geral do conceito empregado e da técnica proposta, além de um exemplo de aplicação.

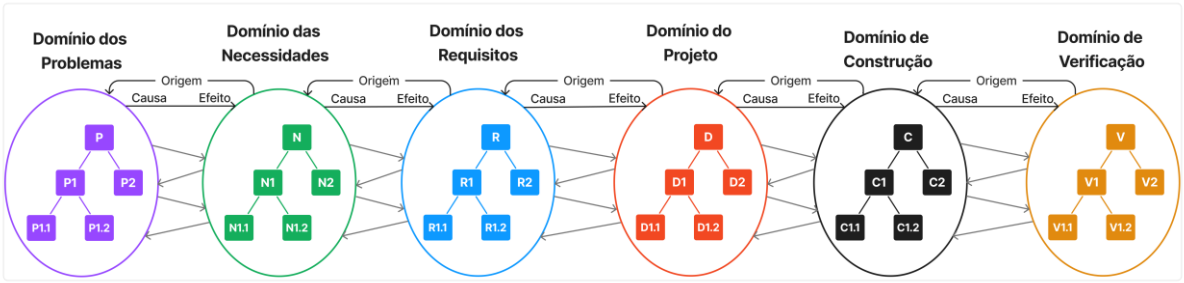
4.1 Visão Geral da Estruturação da Associação entre os Domínios

Nas seções a seguir, será apresentada uma visão geral da estruturação de domínios de especificação e associação. Ademais, será detalhada a associação entre os domínios e realizada uma análise dessas associações.

4.1.1 Estruturação de Domínios de Especificação e Associações

Conforme destacado na seção 2.4 e ilustrado na Figura 7, a abordagem PBSRS propõe que os conceitos relacionados com a atividade de Requisitos de Software sejam organizados nos Domínios de Problemas, Necessidades, Funcional, Físico e Processo (SOUZA e STADZISZ, 2016) (PEREIRA, 2011). Nesta pesquisa de mestrado adaptou-se e expandiu-se a representação de domínios, conforme apresentado na Figura 14. Estas mudanças foram propostas visando deixar a abordagem PBSRS mais bem direcionada ao contexto de Engenharia de Software.

Figura 14 – Ilustração dos domínios



Fonte: Autoria própria (2024)

O **Domínio dos Problemas**, cuja caracterização foi apresentada na seção 3.4, contém o conhecimento das causas originais para o desenvolvimento de um software, ou seja dos interesses, obrigações, expectativas e esperanças que atingem os *stakeholders* e os motivam à busca de uma solução. Por sua vez, o **Domínio das Necessidades** mantém o significado da abordagem PBSRS, ou seja, este domínio contém o conhecimento a respeito daquilo que os *stakeholders*

buscam e que o software deverá prover. A caracterização deste domínio foi feita na seção 3.5. Na Figura 14 percebe-se a relação de causa-efeito do Domínio dos Problemas com o das Necessidades e indicação da relação de origem entre eles. As setas entre os domínios indicam o processo de *zigzaging* (conforme nomenclatura do MIT (SUH, 1990)), ou seja, o processo de iteração na construção do conhecimento nos dois domínios.

O **Domínio Funcional** da PBSRS teve seu nome alterado para o **Domínio dos Requisitos** para fins de clareza (*i.e.*, o termo Funcional poderia remeter a outros entendimentos sobre aspectos de funcionamento do software) e contém o conhecimento a respeito do que o software almejado deverá fazer ou ter como características (também referidas como restrições ou propriedades), visando entregar o que está especificado no Domínio das Necessidades. A caracterização deste domínio foi apresentada na seção 3.6. Na Figura 14, ilustra-se, também, a relação de causa-efeito deste domínio com o das Necessidades, indicando que seu conhecimento é criado como efeito das necessidades especificadas anteriormente. O processo de *zigzaging*, também, se mantém entre estes dois domínios, indicando que os conhecimentos poderiam ser construídos de forma iterativa.

O **Domínio Físico** da PBSRS foi rebatizado para **Domínio de Projeto** e corresponde ao conhecimento associado à concepção ou projeto da solução de software, envolvendo a especificação e projeto técnico de sua arquitetura e comportamento. O **Domínio do Processo** da PBSRS foi fragmentado no **Domínio de Construção** e **Domínio de Verificação** que são domínios de conhecimentos clássicos relacionados a implementação da solução projetada no Domínio de Projeto e aos artefatos de verificação (*e.g.*, testes, revisão e inspeção de software) (SOUZA, 2016) (PEREIRA, 2011). Pertinente para estes três domínios a mesma relação de causa-efeito para com os domínios antecessores e a eventual aplicação do processo de *zigzaging* entre domínios, conforme ilustrado na Figura 14.

A Figura 14 apresenta, assim, uma visão mais alinhada aos conceitos clássicos de processos de desenvolvimento de software em que os quatro domínios mais à direita representam grupos de conhecimentos mais comuns da área, enquanto os dois domínios mais à esquerda representam propostas da PBSRS que estendem a visão clássica. Neste contexto, a abordagem PBSRS se restringe aos três domínios da esquerda que, juntos, especificam os “propósitos” do

desenvolvimento de um software, que compreendem os conhecimentos dos problemas, necessidades e requisitos.

4.1.2 Associação entre os Domínios

Com base na visão dos domínios apresentada na seção anterior, em especial dos Domínios dos Problemas, das Necessidades e dos Requisitos, percebe-se uma relação causal entre os requisitos de software e as necessidades e entre as necessidades e seus problemas. Em outras palavras, os problemas dos *stakeholders* são a causa de suas necessidades, assim como estas necessidades são a causa dos requisitos de software. Por isso, a especificação dos problemas seria realizada antes da especificação das necessidades do cliente e a especificação de requisitos seria realizada após a especificação das necessidades. Nesta ordem, a especificação anterior irá orientar a especificação posterior e, consequentemente, avaliar se cada especificação está completa e correta em relação às suas causas (SOUZA e STADZISZ, 2016).

Em razão disso, a relação entre o **Domínio das Necessidades** (*i.e.*, conjunto das necessidades do cliente para um dado software), denotado **NE**, e o **Domínio dos Requisitos** (*i.e.*, conjunto dos requisitos para um dado software), denotado **RE**, pode ser definida como um subconjunto do produto cartesiano de **NE** e **RE**. Ou seja, a relação, denotada **R1**, é definida como sendo um conjunto de pares ordenados (**ne**, **re**) tais que **ne** pertence ao conjunto **NE** (*i.e.*, Domínio das Necessidades), **re** pertence ao conjunto **RE** (*i.e.*, Domínio dos Requisitos) e **re** contribui para entregar o que **ne** expressa.

$$R1 \subseteq NE \times RE = \{(ne, re) \mid ne \in NE \wedge re \in RE \wedge re\ contribui\ para\ ne\} \quad (1)$$

Para descrever este relacionamento em cada situação da especificação, pode-se empregar uma matriz de incidência, em que as linhas representam as necessidades, as colunas representam os requisitos e cada entrada da matriz define se há uma associação entre a necessidade e o requisito respectivo (assinalado por um “X”). A Tabela II ilustra tal matriz para um caso com quatro necessidades (N01 a N04) e cinco requisitos funcionais (RF01 a RF05) (SOUZA e STADZISZ, 2016) (SUH, 1990). Nota-se que a linha do problema N04, não possui nenhuma assinalação, como será explicado a seguir.

Tabela II - Exemplo de tabela de incidência NE X RE

Necessidades do Cliente (NE)	**Requisitos Funcionais (RE)**				
	RF01	RF02	RF03	RF04	RF05
N01	X				
N02		X	X		
N03			X	X	
N04					

Fonte: Autoria própria (2024)

Por fim, a relação entre o **Domínio dos Problemas** (*i.e.*, conjunto dos problemas dos *stakeholders*), denotado **PO**, e o **Domínio das Necessidades**, denotado **NE**, pode ser definida como um subconjunto do produto cartesiano de **PO** e **NE**. Ou seja, a relação, denotada **R2**, é definida como sendo um conjunto de pares ordenados (**po**, **ne**) tais que **po** pertence ao conjunto **PO** (*i.e.*, Domínio dos Problemas), **ne** pertence ao conjunto **NE** (*i.e.*, Domínio das Necessidades) e **ne** contribui para resolver o que **po** expressa.

$$R2 \subseteq PO \times NE = \{(po, ne) \mid po \in Po \wedge ne \in NE \wedge ne\, contribui\, para\, po\} \quad (1)$$

A Tabela III ilustra um exemplo de matriz de incidência de especificação em que as linhas representam três problemas (P01 a P05) e as colunas representam quatro necessidades (N01 a N05). As associações entre os problemas e as necessidades do cliente são assinaladas por um “X”, indicando quais necessidades colaboram para a resolução de quais problemas. Nota-se que a linha do problema P05, não possui nenhuma assinalação, como será explicado a seguir.

Tabela III - Exemplo de tabela de incidência PO X NE

Problema (PO)	**Necessidade do Cliente (NE)**				
	N01	N02	N03	N04	N05
P01	X				
P02		X	X		
P03			X	X	
P04				X	
P05					

Fonte: R. Autoria própria (2024)

4.1.3 Análise das Associações entre Domínios

De acordo com a abordagem PBSRS, para que o software atenda da melhor forma as exigências dos *stakeholders*, a especificação de requisitos deverá estar completa e correta em relação a especificação de necessidades e de problemas (SOUZA e STADZISZ, 2016). Para verificar se a especificação está correta e completa pode-se realizar uma análise de cobertura das especificações, analisando-se as associações estabelecidas entre os Domínios dos Problemas, da Necessidades e dos Requisitos, conforme discutido a seguir.

- Especificação Completa

Quando se faz referência a uma especificação de requisitos como sendo “completa” pretende-se indicar que nada foi esquecido ou nada está faltando na especificação (IEEE, 1998). Dito de outra forma, deseja-se assegurar que os requisitos especificados contemplam ou cobrem todas as “intenções” ou “demandas” dos *stakeholders*, pois se alguma delas não fosse contemplada, entendería-se que a especificação estaria incompleta, levando a um futuro produto que não entrega tudo que é esperado e que não satisfaria plenamente os clientes.

Uma especificação de necessidades é dita “completa” se, para todo problema contido na Especificação de Problemas, há, pelo menos, uma necessidade na especificação de necessidades que proveja algum entregável que contribui para sua minimização ou resolução. De forma semelhante, uma especificação de requisitos é dita “completa” se, para toda necessidade na Especificação de Necessidades, há, pelo menos, um requisito na Especificação de Requisitos que contribua para a satisfação daquela necessidade.

Como exemplo, a Tabela III mostra uma associação entre problemas e necessidades que pode ser considerada incompleta uma vez que o problema P05 não possui nenhuma necessidade associada. Assim, este problema não estaria sendo considerado na futura solução, que não proveria nenhuma colaboração para a sua resolução. De forma análoga, a Tabela II mostra uma associação incompleta entre necessidades e requisitos em que a necessidade N04 não possui nenhum requisito associado. Isto significa que a solução expressa pelos requisitos não irá prover nenhuma funcionalidade ou propriedade visando a satisfação desta necessidade.

- Especificação Correta

Uma Especificação de Necessidades é dita “correta” se para toda necessidade especificada há, pelo menos, um problema dentro da Especificação de Problemas para qual esta necessidade provê algum entregável que contribua para a minimização ou solução daquele problema. De forma análoga, uma Especificação de Requisito é dita “correta” se para todo requisito especificado há, pelo menos, uma necessidade para a qual o requisito contribui para sua satisfação. Pertinente ressaltar, entretanto, que alguns requisitos de software se originam de necessidades técnicas das soluções (*e.g.*, procedimentos internos de segurança ou qualidade) e, assim, seria justificável que estes não provejam entregáveis necessários à resolução de problemas do cliente.

Por exemplo, a Tabela II apresenta uma associação incorreta em razão de RF05 não estar associado a nenhuma necessidade e, portanto, não contribuir para a solução dos problemas definidos. O mesmo acontece na Tabela III, uma vez que N05 não está associada a nenhum problema. Em outras palavras, N05 irá prover algo para o *stakeholder* que não irá colaborar com nenhum problema definido.

- Interdependências entre Necessidades

As especificações de necessidades podem gerar interdependências ou interferências entre elas, assim como ocorre no projeto técnico de produtos a partir de suas especificações funcionais, estudado por Nam P. Suh (1990) em seus trabalhos sobre Projeto Axiomático, no MIT. Naquele estudo, considera-se que “projeto ideais” são aqueles em que há independência funcional dos parâmetros de projetos com relação aos requisitos funcionais, ou seja, cada parâmetro de projeto

(*i.e.*, a definição de como uma parte da solução atende a algum requisito funcional) afeta apenas o requisito funcional para o qual foi concebido, sem gerar efeitos (ou interferência) contra outros requisitos. Assim, cada requisito funcional é atendido por um ou mais parâmetros de projeto com independência dos demais parâmetros que definem a solução.

Na Tabela II, percebe-se que RF03 contribui para prover alguma ação tanto para N02 quanto para N03. Nesta situação, poderia existir uma interferência entre estas necessidades pela ação de RF03. Na mesma tabela, percebe-se que RF02 e RF03 provêm ações para N02, o que poderia ser considerada uma redundância. Este tipo de análise de interdependências entre necessidades dos *stakeholders* é um campo de estudo ainda em aberto. Esta análise poderia fornecer alguma medida sobre a qualidade da especificação, como aquelas apontadas por Nam Suh nas pesquisas sobre Projeto Axiomático. Entretanto, a autora não encontrou referências a respeito e esta temática não foi incluída no escopo deste trabalho de mestrado, ficando como sugestão para trabalhos futuros.

4.2 Técnica de Especificação de Necessidades Proposta

Nesta seção, será apresentada uma visão geral da atividade de especificação de necessidades do cliente e feita a descrição da técnica proposta de especificação de necessidades. Um exemplo de aplicação da técnica proposta será também apresentado ao final da seção.

4.2.1 Visão Geral da Especificação de Necessidades do Cliente

A atividade de Especificação de Necessidades do Cliente é uma das atividades dentro da abordagem PBSRS (conforme Figura 8). Ela é realizada a partir de dois conhecimentos: a Especificação dos Problemas e o *Glance* da Solução (SOUZA e STADZISZ, 2016). A Especificação dos Problemas é o conhecimento do Domínio dos Problemas (conforme seção 3.5), enquanto o *Glance* da Solução é o conhecimento de mais alto nível relativo à solução almejada (conforme seção 3.5).

Conforme discutido na seção 3.1.1, a especificação de cada necessidade do cliente expressa o que lhe falta e que deverá ser provido pelo software. Assim, durante a especificação de cada necessidade do cliente, deve-se estabelecer o “entregável” que caracteriza a “falta” percebida pelos *stakeholders* com relação a um

ou mais problemas. Um “entregável” é algo que a solução de software poderia prover aos *stakeholders* visando minimizar ou resolver um ou mais problemas. Pode-se considerar as categorias sugeridas anteriormente neste documento (seção 3.1.2) como referência para expressar os entregáveis nas especificações de necessidades.

A Especificação de Necessidades do Cliente é uma atividade a ser realizada pelo especialista em requisitos que envolve, ao mesmo tempo: (i) uma prospecção junto aos *stakeholders* de sua percepção de faltas que possam ser providas por uma solução de software e (ii) uma proposição de “entregáveis” que a solução de software possa prover em vista dos problemas contidos na Especificação de Problemas, das faltas percebidas pelos *stakeholders* e da especificação do *Glance* da solução. A expressão da necessidade é alcançada com a reunião da falta e do que a solução deverá prover.

Como exemplo, considere o problema “O médico deve fornecer o atestado em papel ou digital quando solicitado pelo paciente, sob pena de descumprimento de regulamentação médica”. O médico, orientado por algum profissional, idealizou uma solução para este e, eventualmente, outros problemas seus, na forma de um Sistema de Informação de Suporte a Consultas, denominado SISC. Esta idealização do SISC corresponde ao *Glance* da Solução. Na especificação de necessidades relativas a este problema, o especialista pode, como primeiro passo, prospectar junto ao médico o que lhe falta para resolver ou minimizar este problema por meio do SISC. Eventualmente, esta prospecção poderia concluir pela falta de capacidade (*i.e.*, de meios) do médico para criar atestados digitais. Como segundo passo, o especialista poderia propor que o SISC pudesse prover ao médico a capacidade de criar atestados digitais. Assim, a necessidade identificada, que reúne a falta percebida e o que a solução deveria prover, poderia ser relatada como: “O médico precisa do SISC para ter capacidade de criar atestados digitais.”. Lembrar que “criar atestados” não descreve uma funcionalidade do futuro software e sim o que ele deve prover que é a “capacidade” para criar os atestados. Seguramente, os requisitos que se desdobrarão desta necessidade indicarão exigências funcionais e não funcionais para o software pelo meio das quais ele poderá prover a criação de atestados.

Ao longo desta atividade de especificação, o especialista em requisitos deve estar atento às nuances e variações nas percepções de faltas que podem ser providas por um software expressas pelo *stakeholder*. Isso inclui identificar diferentes faltas percebidas e como estas faltas podem ser abordadas pela solução

de software. A capacidade de capturar estas faltas e identificar uma solução do que o software poderá prover de maneira precisa e compreensível é crucial para o desenvolvimento de um software que realmente atendas as necessidades do cliente. Entretando, esta atividade pode não ser simples, pois, de certa forma, se está produzindo uma concepção da solução em termos de seus propósitos e visão funcional. O exemplo anterior foi um tanto óbvio na identificação da falta, do entregável e da reunião dos dois na definição da necessidade, outras situações podem requerer uma reflexão maior para bem compor as especificações das necessidades.

Deve-se considerar, no mapeamento dos problemas em necessidades, que uma necessidade poderia especificar entregáveis que a solução de software poderia prover satisfazendo, um determinado problema ou, até, mais de um problema em diferentes graus. Também, é possível que mais de uma necessidade proveja entregáveis para um mesmo problema. O mesmo procede para o mapeamento entre as especificações de necessidades e requisitos do software.

Como exemplo, consideremos dois problemas (P1 e P2) relacionados com o trânsito em uma cidade, conforme definido a seguir.

- P1 – A prefeitura deve auxiliar a reduzir os atropelamentos na cidade, sob pena de reclamações e processos das famílias e pessoas atingidas.
- P2 – A prefeitura deve indicar aos veículos o local de parada antes dos semáforos, sob pena de reclamação dos pedestres.

Visando prover soluções para estes problemas, seria possível imaginar duas necessidades para o problema P1.

- N1 – A prefeitura precisa de um sistema para automatizar a sinalização de avanço dos pedestres.
- N2 – A prefeitura necessita de um sistema de sinalização no chão (faixas) para orientar os locais seguros de travessia.

Para a solução do problema P2, seria possível identificar a seguinte necessidade.

- N3 – A prefeitura necessidade de um sistema de sinalização no chão (faixas) para indicar o limite de parada dos veículos.

Nota-se, neste exemplo, que N2 e N3 são equivalentes, pois ambas remetem a necessidade de informar, por meio de faixas, o local de passagem dos

pedestres, que, ao mesmo tempo, é o limite de parada dos veículos. Assim, N2 e N3 poderiam ser consideradas a mesma necessidade, contribuindo para a solução de dois problemas.

A especificação de necessidades poderia ser conduzida de três formas, conforme descrito a seguir.

- **Abordagem Sequencial:** neste caso, após a conclusão integral de toda a Especificação dos Problemas e da elaboração do *Glance*, produz-se a especificação integral de todas as necessidades do cliente. Por exemplo, no exemplo da Tabela III da subseção 4.1.2, a especificação dos problemas P01, P02, P03, P04 e P05 poderia ter sido feita inteiramente e, então, a especificação das necessidades N01, N02, N03, N04 e N05 poderia ter sido realizada em função do conjunto total de problemas. Nesta abordagem, em um primeiro momento, a atenção está toda focada em entender e especificar as obrigações e penalizações (*i.e.*, problemas) que recaem sobre os *stakeholders* e que constituem as motivações para a busca por uma solução. A partir dos problemas, elabora-se o *Glance*, que é o esboço da solução almejada. Em um próximo momento, o especialista se concentra em determinar todas as necessidades requeridas para resolver ou minimizar os problemas por meio da solução esboçada, focando em todos os entregáveis necessários. Como, eventualmente, uma necessidade pode estar associada a mais de um problema e mais de uma necessidade pode ser associada a um mesmo problema, a especificação prévia de todos os problemas, facilita a especificação das necessidades.
- **Abordagem Gradual:** neste caso, após a definição de cada problema na Especificação de Problemas, já se produz a especificação da ou das necessidades correspondentes e depois retorna-se à especificação do próximo problema. Como exemplo, ainda na Tabela III, após finalizar a especificação do problema P01, já poderia ser realizada a especificação da necessidade que irá atender este problema, que na tabela seria somente a N01. Após finalizar a especificação de N01, o especialista volta a verificar se existem mais problemas para serem especificados. Nesta abordagem, a especificação das necessidades ocorre imediatamente após e identificação e especificação de cada problema, permitindo manter o foco naquele problema e nas necessidades decorrentes. Entretanto, a idealização do *Glance* deveria ser feita também de forma gradual, o que pode trazer dificuldades e, até, inviabilizar esta abordagem.

- **Abordagem Iterativa:** neste caso, a especificação de problemas, a elaboração do *Glance* e a especificação de necessidades são realizadas por meio de iterações, ou seja, ciclos de refinamentos envolvendo o refinamento da especificação de problemas, do *Glance* e da especificação das necessidades. A cada ciclo, um ou mais problemas podem ser identificados e especificados, o *Glance* pode ser refinado e uma ou mais necessidades podem ser definidas e associadas aos problemas. Assim, a cada iteração, produz-se uma visão incompleta do conjunto da especificação dos problemas e necessidades e do *Glance*. Considerando o exemplo da Tabela III, a primeira interação poderia envolver a especificação dos problemas P1 e P2, pois eles poderiam ter alguma relação de proximidade. Ainda nesta iteração, poderia ser gerada uma primeira versão do *Glance* (*i.e.*, versão G1) e especificadas as necessidades N1 e N2. Desta forma, (P1, P2, G1, N1 e N2) constituiriam o produto da primeira iteração. Na segunda iteração poderia ser acrescentada a especificação de um terceiro problema (P3), ser refinado o *Glance* (G2) e especificadas as necessidades N3 e N4), levando ao produto (P1, P2, P3, G2, N1, N2, N3 e N4), e assim por diante nas próximas iterações.

De forma análoga, a especificação de requisitos pode ser elaborada de forma Sequencial, Gradual ou Iterativa com relação à Especificação de Necessidades. Assim, na Abordagem Sequencial, seriam especificados todos os problemas, primeiramente. A seguir, seriam especificadas todas as necessidades e, por fim, todos os requisitos associados às necessidades. Na Abordagem Gradual, seria especificado um problema por vez, seguido da especificação das necessidades associadas e, então, dos requisitos associados a estas necessidades. Por fim, na Abordagem Iterativa, as necessidades e requisitos seriam desenvolvidos em conjunto por meio de uma sucessão de ciclos (*i.e.*, iterações) de refinamento. Alternativamente, estas três abordagens de especificação poderiam ser aplicadas às três especificações (*i.e.*, problemas, necessidades e requisitos) em conjunto.

Em outras palavras, esta associação de identificação das faltas e proposição dos entregáveis, independente da forma como seja conduzida a iteração e colaboração com os *stakeholders*, permite que o especialista compreenda mais profundamente os problemas e as faltas percebidas pelos *stakeholders*. Por meio de uma prospecção detalhada e da proposição de entregáveis claros e viáveis, é possível realizar a atividade de Requisito de Software de maneira mais sistemática,

resultando em um software que efetivamente solucione os problemas dos *stakeholders*.

4.2.2 Descrição da Técnica Proposta

Considerando a estruturação e associação de domínios e a visão geral da atividade de Especificação de Necessidades descritas nas seções anteriores deste capítulo, descreve-se nesta seção a Técnica Proposta para especificar as necessidades do cliente, alinhada com a abordagem PBSRS. A Figura 15 apresenta uma ilustração geral da técnica proposta para especificação de necessidades do cliente. As atividades envolvidas são descritas a seguir.

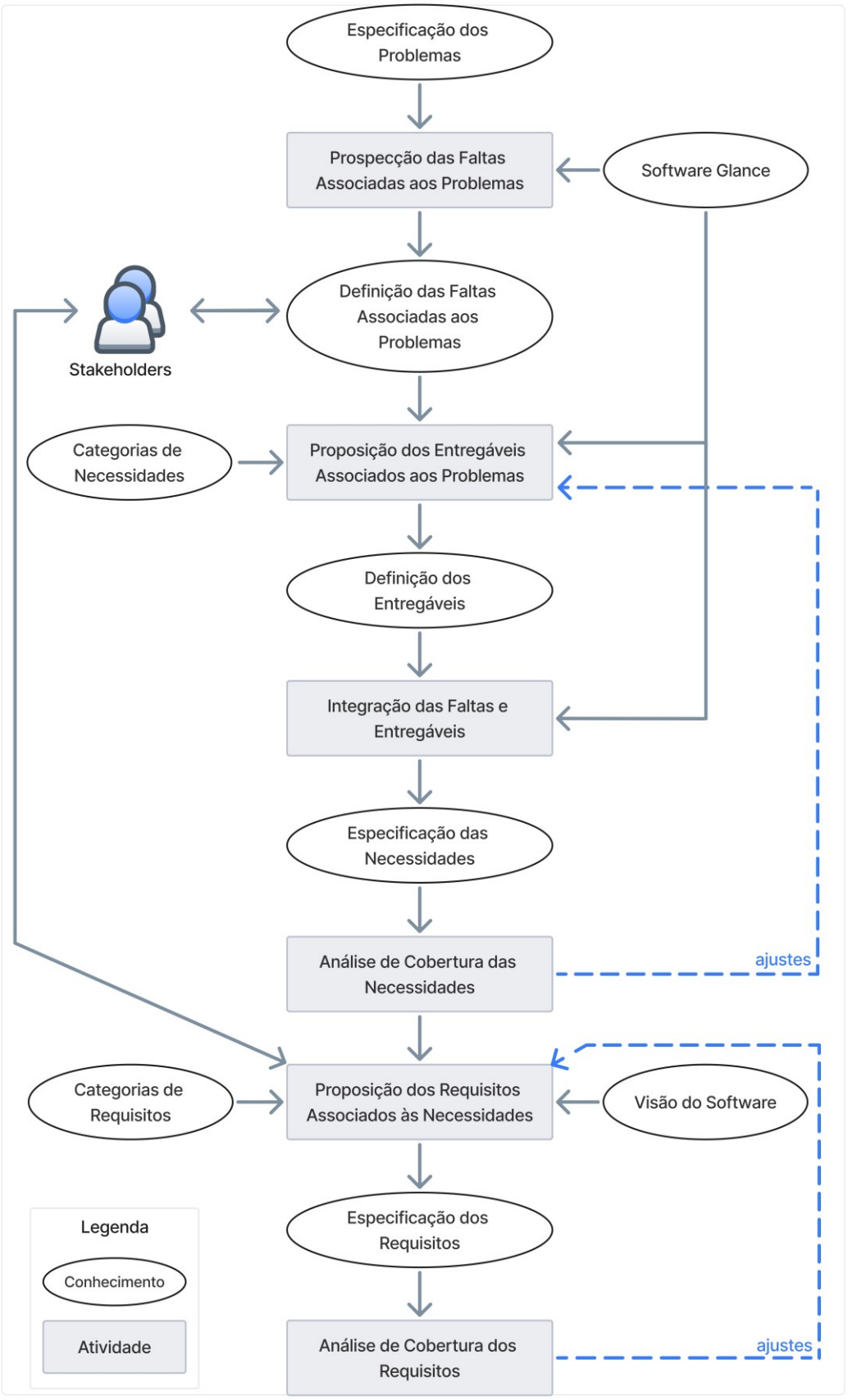
- **Prospecção das Faltas Associadas aos Problemas**

Na primeira atividade, denominada “Prospecção das Faltas Associadas aos Problemas”, o especialista em requisitos irá interagir com os *stakeholders* visando levantar quais são as faltas sentidas por eles em relação ao conjunto de problemas definidos (*i.e.*, o que os *stakeholders* sentem que lhes falta para minimizar ou resolver seus problemas). Neste estágio, o *Glance* da solução já estaria totalmente ou parcialmente estabelecido e deveria ser considerado nesta atividade de prospecção das faltas. O *Glance* pode limitar as escolhas da definição das faltas uma vez que ele define o esboço da solução e as faltas devem ser compatíveis com o caminho de solução escolhido. Ou seja, para cada falta que for definida, deve-se checar sua consistência com o *Glance*.

- **Proposição dos Entregáveis Associados aos Problemas**

Na segunda atividade, denominada “Proposição dos Entregáveis Associados aos Problemas”, o especialista em requisitos irá propor entregáveis que o software (*i.e.*, *Glance*) poderá prover e que respondam às faltas definidas juntos aos *stakeholders*. No caso ideal, a especificação de necessidades contém um entregável que provê diretamente o que é descrito por cada falta. Entretanto, pode ser necessário ao especialista adequar os entregáveis ou adequar as faltas para que se alcance um entregável viável para cumprir o que se pretende com uma falta. Neste sentido, a visão geral da solução de software contida no *Glance* servirá de base para avaliar a viabilidade e compatibilidade dos entregáveis.

Figura 15 – Visão geral da técnica proposta



Fonte: Autoria própria (2024)

- **Integração das Faltas e dos Entregáveis**

Na terceira atividade, denominada “Integração das Faltas e dos Entregáveis”, o especialista em requisitos irá compor a expressão das necessidades do cliente, de acordo com os conceitos e formato sugerido na seção 3.6, reunindo as definições de faltas e entregáveis definidas nas duas atividades anteriores. A expressão do conjunto de necessidades compõe a Especificação das Necessidades do Cliente.

- **Análise de Cobertura das Necessidades**

Na quarta atividade, o especialista em requisitos irá realizar a “Análise de Cobertura das Necessidades” visando averiguar se a especificação está correta e completa, conforme discutido na seção 4.1.3. Para tanto, deve-se criar a Matriz de Associações que indica quais necessidades participam da resolução ou minimização de cada problema definido. Outras análises podem, também, ser conduzidas, como a análise de interdependências funcionais e de complexidade da especificação, conforme discutido na seção 4.1.3.

- **Proposição dos Requisitos Associados às Necessidades**

Na quinta atividade, por sua vez, denominada “Proposição dos Requisitos Associados às Necessidades”, o especialista em requisitos irá propor um ou mais requisitos que satisfaçam cada uma das necessidades, ou seja, requisitos funcionais e não funcionais que produzam os entregáveis previstos nas necessidades.

- **Análise de Cobertura dos Requisitos**

Por fim, na atividade de “Análise de Cobertura dos Requisitos”, o especialista em requisitos irá averiguar se a especificação de requisitos está correta e completa frente ao conjunto de necessidades. Da mesma forma que para as necessidades, pode-se também realizar outras análises como a de interdependências funcionais e de complexidade da especificação de requisitos.

4.2.3 Exemplo de Aplicação da Técnica Proposta

Para exemplificação da técnica proposta, considere-se, como exemplo, uma granja em que é feita a criação de aves para comercialização. Certamente, o

produtor enfrenta diversos problemas nesta atividade pecuária, entre eles, por exemplo, a obrigação de controlar a iluminação no ambiente de criação, que tem efeitos sobre o crescimento e o bem-estar das aves. Desta forma, poderia ser definido, hipoteticamente, o seguinte problema da granja:

P1 – A granja deve assegurar uma iluminação apropriada no ambiente de criação das aves, sob pena de perda de produtividade e de redução do bem-estar das aves.

Observando este problema, os *stakeholders* e/ou a equipe de projeto poderiam imaginar um vislumbre da solução (*i.e.*, um *Glance*) com vista a alcançar um sistema que pudesse resolver ou minimizar este problema. Para o exemplo considerado, o *Glance* poderia ser:

***Glance* – A solução imaginada seria um sistema computacional de controle de iluminação que consulta uma tabela de configuração e regula automaticamente a iluminação do ambiente conforme os parâmetros da tabela. Esta tabela considera a idade das aves dentro do ciclo de criação e os horários de amanhecer e anoitecer. O sistema deverá considerar, também, o grau de iluminação natural no ambiente, de maneira a prover apenas a complementação de iluminação necessária para atingir os níveis desejados.**

A especificação de Necessidades do *stakeholder*, partindo dos problemas e do *Glance*, irá, então, prospectar as “faltas” dos *stakeholders*, visando a solução de seus problemas. Esse trabalho, seguramente, requer competência do especialista em requisitos, pois envolve uma interação com os *stakeholders*, diagnóstico dos problemas e observação do contexto de negócio, além da consideração do *Glance* da solução. Para o exemplo considerado, o especialista em requisitos poderia ter prospectado as seguintes “faltas”:

F1 – Falta de conhecimento sobre a iluminação artificial adicional requerida ao longo do dia para cada ambiente de criação de frangos da granja.
F2 – Falta de capacidade de ação sobre a regulagem do fornecimento de iluminação artificial em cada ambiente de criação de frangos da granja.

Em conjunto com esta prospecção de faltas, o especialista deve identificar ou conceber os entregáveis respectivos do software. Em outras palavras, ele deve elaborar uma ou mais especificações de provimento que atendam as faltas prospectadas, por meio do que a solução poderia entregar aos *stakeholders*. Por exemplo, considerando as faltas F1 e F2, os entregáveis poderiam ser:

E1 – Conhecimento da iluminação artificial adicional requerida ao longo do dia para cada ambiente de criação de frangos da granja.
E2 – Controle de ações sobre a regulagem do fornecimento de iluminação artificial em cada ambiente de criação de frangos da granja.

Após documentar os entregáveis identificados, o especialista elabora a especificação de necessidades do *stakeholder* pela integração das faltas e entregáveis relacionados. Para as faltas F1 e F2 e entregáveis E1 e E2, o resultado poderia ser:

N1 – A granja precisa de um sistema para ter conhecimento da iluminação artificial adicional requerida ao longo do dia para cada ambiente de criação de frangos da granja.
N2 – A granja precisa de um sistema para controlar as ações sobre a regulagem do fornecimento de iluminação artificial em cada ambiente de criação de frangos da granja.

A partir da especificação criada, pode-se realizar a Análise de Cobertura da especificação. No exemplo considerado, ou neste estágio do exemplo, a análise de cobertura de necessidades e problemas não procede, pois há apenas um problema e, obviamente, todas as necessidades se aplicam a atender este problema. Assim, a matriz tomaria a forma da tabela a seguir, contendo apenas uma linha, relacionada

ao problema P1, concluindo-se que, até este ponto, a especificação é completa e correta.

Tabela IV - Tabela de incidência PO X NE

Problema (PO)	**Necessidade do Cliente (NE)**	
	N01	N02
P01	X	X

Fonte: Autoria própria (2024)

5 CASO DE ESTUDO

Neste capítulo é apresentado um caso de estudo relacionado a uma empresa de comércio eletrônico de mídia digitais que foi desenvolvido no contexto desta pesquisa de mestrado. O objetivo com este caso de estudo foi aplicar, na prática, a técnica e os conceitos sugeridos visando obter um diagnóstico sobre a sua utilização.

5.1 Visão Geral do Negócio

O objeto de estudo é um software *web* que apresenta um portal (*i.e.*, *marketplace*) para venda de conteúdos digitais desenvolvidos por terceiros, denominados “produtores”. Por meio deste portal, os consumidores (*i.e.*, compradores e clientes dos conteúdos) poderão consultar as ofertas de conteúdos e fazer a aquisição de licenças de acesso. Trata-se de um novo produto real, em desenvolvimento por uma empresa brasileira, para o qual estuda-se sua especificação de requisitos. Ressalta-se que alguns dados a respeito do produto e do negócio foram removidos visando preservar o sigilo de informações da empresa, inclusive sua identificação. Além disso, este caso de estudo não é completo, ou seja, ele aborda apenas uma parcela do software pretendido pela empresa.

5.1.1 Visão Geral do Contexto de Negócio

Embora a empresa fornecedora já atue no mercado de venda de conteúdos digitais, seu objetivo estratégico atual é o de realizar uma mudança de foco e atividades, direcionando maior atenção e recursos aos produtores iniciantes de conteúdo. Essa iniciativa visa proporcionar suporte especializado, oferecendo condições favoráveis para o crescimento e desenvolvimento desses produtores. Assim, a empresa busca disponibilizar ferramentas, informações e serviços que sejam particularmente úteis para os produtores que estão começando no mercado. A empresa, também, disponibilizará recursos essenciais, como plataformas tecnológicas, materiais educacionais e oportunidade de *networking*, e criação de um ecossistema que estimule a criatividade e a experimentação, permitindo que os produtores iniciantes se destaquem no mercado.

Esta mudança de objetivo estratégico da empresa também visa atender melhor os consumidores de conteúdo, tanto Pessoas Físicas, quando Pessoas

Jurídicas. Desta forma, a empresa espera proporcionar aos consumidores de conteúdo uma gama mais ampla e diversificada de materiais de alta qualidade, melhorar a fidelidade e a satisfação dos consumidores por meio de uma oferta contínua de conteúdo relevante, e assegurar que os consumidores de conteúdo tenham acesso a produtores que trazem novas perspectivas e ideias, mantendo-os competitivos e atualizados com as tendências de mercado.

Este caso de estudo foca no desenvolvimento de um software de *marketplace* para a venda de conteúdos digitais de produtores iniciantes, conforme descrito nas subseções a seguir.

5.1.2 Definição dos Stakeholders

Neste caso de estudo, decidiu-se por considerar apenas um *stakeholder* que é a Empresa Fornecedora de Conteúdos Digitais (*i.e.*, Fantasia Company) que é a empresa compradora do software a ser desenvolvido. Esta limitação de escolha dos *stakeholders* teve por objetivo manter o caso de estudo mais focado na empresa fornecedora e mais reduzido. Para identificar seus interesses, foram feitas entrevistas que seus potenciais clientes que seriam os Produtores Iniciantes de Conteúdo, Consumidores Pessoa Física e Consumidores Pessoa Jurídica.

Os Produtores Iniciantes de Conteúdo buscam fazer as primeiras vendas com apoio de estratégias de *marketing* e de venda eficazes e maximizar o alcance e a comercialização de seus conteúdos. Assim, eles poderiam trabalhar e efetivar sua transição de carreira para produtor de conteúdos digitais.

Os Consumidores Pessoa Física buscam informações atualizadas sobre as últimas tendências em suas áreas de atuação para manterem-se competitivos, terem dados e *insights* sobre os produtos digitais mais populares e bem-sucedidos dentro de seu nicho de atuação e terem conteúdo detalhado e relevante sobre os produtos digitais disponíveis, ajudando na tomada de decisão sobre a compra de conteúdos digitais.

Por fim, os Consumidores Pessoa Jurídica buscam obter conhecimento por meio de informações detalhadas sobre os principais produtores de conteúdo, suas qualificações e áreas de especialização, e avaliações e testemunhos de outras empresas que utilizam os treinamentos dos produtores, permitindo uma escolha informada e segura.

Descrevem-se, a seguir, os interesses do *stakeholder* Empresa Fornecedora.

Nome	Fantasia Company
Descrição	Este *stakeholder* representa a empresa fornecedora do futuro *marketplace* e que é o "cliente comprador" do software.
Classificação	Cliente Institucional
Interesses	• **I01:** Importância: alta / Objeto: ofertar meios de venda digital visando auxiliar os Produtores Iniciantes alcançarem a rentabilidade desejada. • **I02:** Importância: alta / Objeto: promover as primeiras vendas dos Produtores Iniciantes para auxiliá-los a realizarem a transição de carreira para produtor de conteúdo digital. • **I03:** Importância: alta / Objeto: aumentar a atratividade dos conteúdos digitais. • **I04:** Importância: alta / Objeto: aumentar a confiança na qualidade dos conteúdos digitais.

5.2 Especificação de Problemas

A partir da definição do *stakeholder* e de seus interesses, parte-se para a especificação dos problemas do *stakeholder*. Como discutido na seção 3.4, se um *stakeholder* tem interesses, então considera-se que ele possui "problemas" relacionados. As especificações dos problemas da Fantasia Company, considerando os Produtores Iniciantes de conteúdo são:

Interesse I01: *ofertar meios de venda*

- **P01:** A Fantasia Company deve ofertar meios de venda digital, sob pena de não atração e de não alcance de rentabilidade desejado pelos produtores de conteúdo.

Interesse I02: *promover as primeiras vendas*

- **P02:** A Fantasia Company deve ofertar meios para promover as primeiras vendas dos produtores iniciantes, sob pena de desapontamento e desistência de suas tentativas de tornarem-se produtores de conteúdo e de perda de clientes produtores de conteúdo.

Interesse I03: *aumentar a atratividade*

- **P03:** A Fantasia Company deve assegurar a atratividade das ofertas de conteúdos digitais, sob pena de não conquistar os clientes compradores de conteúdos digitais.

Interesse I04: *aumentar a confiança na qualidade*

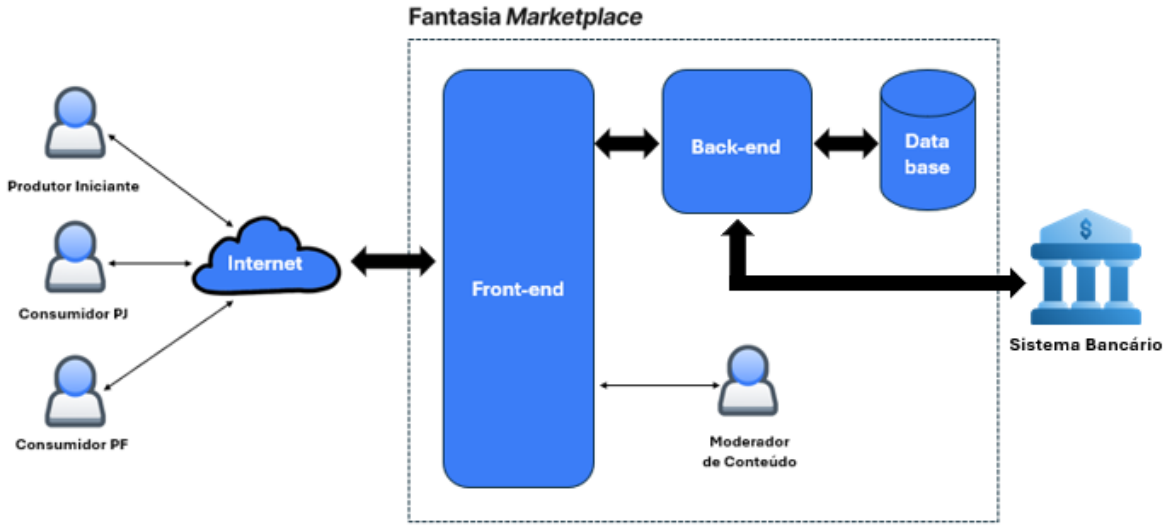
- **P04:** A Fantasia Company deve garantir uma percepção de confiança aos consumidores com relação aos conteúdos digitais, sob pena de não conquistar os clientes compradores de conteúdo digitais.

5.3 Glance

A Fantasia Company vislumbra, como solução para os problemas especificados na seção 5.1.3, um *marketplace* de conteúdo digital para promover as vendas de produtores iniciantes. Este *marketplace* tem como objetivo prover um meio de venda de conteúdos digitais, facilitar as primeiras vendas dos produtores iniciantes e auxiliar os consumidores a encontrarem conteúdos digitais desejados.

A figura a seguir ilustra o *Glance* do *marketplace*, com seus usuários e componentes principais. O *marketplace* será uma solução *web* por meio da qual os consumidores, tanto pessoa física quanto jurídica, poderão visualizar e adquirir conteúdos digitais de produtores iniciantes. Para isso, o produtor iniciante deverá se cadastrar na plataforma e registrar seu conteúdo digital, que só será exibido no *marketplace* após a aprovação de um moderador de conteúdo. O sistema será estruturado de forma que os produtores iniciantes e consumidores se conectarão via *internet* ao *front-end*, a interface do usuário. O *front-end*, por sua vez, se comunicará com o *back-end*, que processa as solicitações, aplica a lógica de negócios e acessa o banco de dados para armazenar e recuperar informações. Após a aprovação pelo moderador, o conteúdo será disponibilizado aos consumidores no *marketplace* com todas as características informadas durante o registro. Caso o consumidor deseje comprar o conteúdo digital, ele precisará estar cadastrado na plataforma e o *back-end* se encarregará de processar a transação financeira através da integração com o sistema bancário, completando o fluxo operacional do *marketplace*.

Figura 16 – *Glance* do *marketplace*



Fonte: Autoria própria (2024)

5.4 Especificação das Faltas

As faltas percebidas de acordo com o diagnóstico dos problemas feito pela autora, no papel de especialista em requisitos, e considerando a observação do contexto do negócio e do *Glance* da solução estão relacionadas a seguir.

Problema P01: *ofertar meios de venda digital*

- **F01:** Falta de capacidade da Fantasia Company para criar um acervo de produtores e conteúdos digitais para venda.
- **F02:** Falta de conhecimento da Fantasia Company em ofertar os conteúdos digitais dos Produtores Iniciantes.
- **F03:** Falta de capacidade da Fantasia Company em efetuar automaticamente a venda dos conteúdos digitais dos Produtores Iniciantes.

Problema P02: *ofertar meios para promover as primeiras vendas*

- **F04:** Falta de capacidade da Fantasia Company para promover as primeiras vendas dos conteúdos digitais dos Produtores Iniciantes.

Problema P03: *assegurar a atratividade das ofertas*

- **F05:** Falta de conhecimento da Fantasia Company sobre quais conteúdos digitais seriam mais atrativos segundo o perfil dos consumidores.

- **F06:** Falta de capacidade da Fantasia Company em criar atrativos para conteúdos digitais em destaque.

Problema P04: *fornecer uma percepção de confiança*

- **F07:** Falta de capacidade da Fantasia Company em fornecer uma percepção de confiança aos consumidores com relação aos conteúdos digitais.

5.5 Especificação dos Entregáveis

Para realizar a especificação dos entregáveis, o especialista em requisitos deve analisar o conjunto das faltas percebidas e identificar o tipo da necessidade, conforme apresentado na seção 3.6. A seguir é apresentada a especificação dos entregáveis associados a cada falta de cada problema.

Problema P01 / Falta F01:

- **E01:** Criação de um acervo de produtores e conteúdos digitais.

Problema P01 / Falta F02:

- **E02:** Informação sobre as ofertas de conteúdos digitais dos Produtores Iniciantes.

Problema P01 / Falta F03:

- **E03:** Automatismo de venda de conteúdos digitais dos Produtores Iniciantes.

Problema P02 / Falta F04:

- **E04:** Automatismo para promover as primeiras vendas dos conteúdos digitais dos Produtores Iniciantes.

Problema P03 / Falta F05:

- **E05:** Informação sobre conteúdos atrativos segundo o perfil de cada consumidor.

Problema P03 / Falta F06:

- **E06:** Automatismo de geração de atrativos para conteúdos digitais.

Problema P04 / Falta F07:

- **E07:** Informação relevantes sobre os conteúdos digitais.

5.6 Especificação de Necessidades

A especificação das necessidades é feita pela integração da especificação das faltas e dos entregáveis respectivos. A especificação das necessidades da Fantasia Company para o caso de uso está descrita a seguir.

Problema P01: *ofertar meios de venda digital*

- **N01:** A Fantasia Company precisa de um *marketplace* para a criação de um acervo de conteúdo digitais.
- **N02:** A Fantasia Company precisa de um *marketplace* para que seus consumidores tenham conhecimento a respeito dos conteúdos digitais dos Produtores Iniciantes.
- **N03:** A Fantasia Company precisa de um *marketplace* para automatizar as vendas de conteúdos digitais.

Problema P02: *ofertar meios para promover as primeiras vendas*

- **N04:** A Fantasia Company precisa de um *marketplace* para automatizar a promoção das primeiras vendas de conteúdos digitais dos Produtores Iniciantes.

Problema P03: assegurar a atratividade das ofertas

- **N05:** A Fantasia Company deseja um *marketplace* para que seus consumidores tenham conhecimento sobre conteúdos digitais alinhados ao perfil.
- **N06:** A Fantasia Company necessita de um *marketplace* para automatizar a geração de atrativos sobre conteúdos digitais alinhados ao perfil dos clientes compradores.

Problema P04: *fornecer uma percepção de confiança*

- **N07:** A Fantasia Company deseja um *marketplace* para que seus consumidores tenham conhecimento de dados relevantes sobre os conteúdos digitais.

5.7 Análise de Coberturas dos Problemas

A Tabela V apresenta a associação entre os Problemas e as Necessidades do Cliente. Observa-se nesta tabela que não há linhas nem colunas em branco, indicando que a especificação está correta e completa. Observa-se, ainda, que há um número maior de colunas (*i.e.*, seis necessidades) do que de linhas (*i.e.*, quatro problemas). Embora esta condição seja considerada indesejada na teoria de projeto axiomático (SUH, 1990), na abordagem PBSRS ela é considerada comum e aceitável, dado que as necessidades são, por natureza, mais detalhadas, enquanto os problemas são, naturalmente, mais abstratos. Assim, a especificação de mais de uma necessidade para atender a um mesmo problema deverá ocorrer frequentemente quando um problema gera mais de uma falta e a solução provê mais

de um entregável. Assim, do ponto de vista da abordagem PBSRS, apoiada na teoria de projeto axiomático, a Tabela V descreve uma “especificação ideal”.

Tabela V - Tabela Análise de Cobertura de Problemas

Problema	Necessidade do Cliente						
	N01	N02	N03	N04	N05	N06	N07
P01	X	X	X				
P02				X			
P03					X	X	
P04							X

Fonte: Autoria própria (2024)

5.8 Especificação de Requisitos

As especificações de requisitos consideradas nesta seção não representam ainda uma decisão final da empresa interessada, pois este caso de estudo acompanhou a empresa em um período específico de dois meses do desenvolvimento do projeto do *marketplace*. Assim, parte da especificação de requisitos apresentada a seguir, pode não representar a realidade atual do produto desenvolvido ou em desenvolvimento.

Necessidade N01: *criação de um acervo de conteúdos digitais*

- **RF01:** O *marketplace* deverá permitir ao produtor de conteúdo se cadastrar no *marketplace*.
- **RF02:** O *marketplace* deverá permitir ao produtor de conteúdo cadastrar um novo conteúdo digital que deverá ser aprovado pelo moderador de conteúdo.
- **RF03:** O *marketplace* deverá permitir ao moderador de conteúdo aprovar ou recusar um novo conteúdo inserido no *marketplace*.

Necessidade N02: ***conhecimento a respeito dos conteúdos digitais***

- **RF04:** O *marketplace* deverá permitir ao usuário navegar sobre as categorias padrões de conteúdos digitais de seu acervo.
- **RF05:** O marketplace deverá permitir ao usuário visualizar a página de informações do conteúdo digital selecionado.

Necessidade N03: ***automatização das vendas***

- **RF06:** O *marketplace* deverá permitir ao usuário efetuar seu cadastro na plataforma para poder efetuar compras.
- **RF07:** O *marketplace* deverá permitir ao usuário visitante efetuar a autenticação de usuário cadastrado.
- **RF08:** O *marketplace* deverá permitir ao usuário logado selecionar um novo conteúdo digital para o carrinho de compra.
- **RF09:** O *marketplace* deverá permitir ao usuário efetuar o pagamento dos itens em seu carrinho de compra.

Necessidade N04: *automatização da promoção das primeiras vendas*

- **RF10:** O *marketplace* deverá destacar os conteúdos de primeira venda em uma prateleira chamada "*Première*", enquanto não efetuar as suas três primeiras vendas.
- **RF11:** O *marketplace* deverá gerar promoções com descontos em conteúdos digitais até a realização das dez primeiras vendas.

Necessidade N05: *conhecimento sobre conteúdos digitais alinhados ao perfil*

- **RF12:** O *marketplace* deverá apresentar categorias de prateleiras alinhadas ao perfil de cada usuário logado.

Necessidade N06: *automatização da geração de atrativos*

- **RF13:** O *marketplace* deverá notificar, por e-mail, os usuários cadastrados quando o seu produtor favorito adicionar um conteúdo digital.
- **RF14:** O *marketplace* deverá notificar, por e-mail, os usuários cadastrados quando um novo conteúdo digital nas categorias alinhadas ao seu perfil for adicionado ao *marketplace*.
- **RF15:** O *marketplace* deverá permitir a um produtor adquirir destaques pagos em *banner* do *marketplace*.
- **RF16:** O *marketplace* deverá apresentar destaques pagos de conteúdos digitais no *banner* de acordo com o perfil do consumidor.

Necessidade N07: *conhecimento de dados relevantes*

- **RF17:** O *marketplace* deverá permitir ao usuário consultar os produtores mais bem avaliados, conforme a categoria selecionada.
- **RF18:** O *marketplace* deverá permitir ao usuário consultar os conteúdos digitais mais vendidos, conforme categoria selecionada.

5.9 Análise de Cobertura das Necessidades

A Tabela VI apresenta a associação das Necessidades do Cliente e Requisitos, que auxilia na análise de cobertura das Necessidades. De imediato, observa-se que não há linhas ou colunas vazias, indicando que esta especificação está correta e completa.

Tabela VI - Tabela Análise de Cobertura das Necessidades

Requisito	Necessidade do Cliente						
	N01	N02	N03	N04	N05	N06	N07
RF01	X						
RF02	X						
RF03	X						
RF04		X					
RF05		X					
RF06			X				
RF07			X				
RF08			X				
RF09			X				
RF10				X			
RF11				X			
RF12					X		
RF13						X	
RF14						X	
RF15						X	
RF16						X	
RF17							X
RF18							X

Fonte: Autoria própria (2024)

Observa-se, na Tabela VI, que a maioria das associações estão alinhadas próximas da diagonal principal da tabela, demonstrando uma boa independência entre as associações. Segundo a teoria de projeto axiomático (SUH, 1990), isto indicaria que a solução é ou está próxima de uma solução de projeto ideal. Eventualmente, poderia haver associações deslocadas da diagonal principal. Entretanto, este desalinhamento poderia ser devido apenas à ordenação dos requisitos e necessidades. Rearranjando algumas linhas e colunas poderia ser feito um melhor alinhamento das associações com a diagonal principal da tabela.

Por fim, observa-se que as colunas, exceto N05, possuem mais de uma associação e que as linhas possuem apenas uma associação cada. Quando uma linha possui mais de uma associação com diferentes necessidades, isto pode indicar a presença de redundâncias (*i.e.*, necessidades podem estar representando a mesma falta, dita de maneira fragmentada ou em duplicidade) ou de excesso de abstração (*i.e.*, o requisito representa ou abstrai mais de uma exigência funcional). Por outro lado, esta situação pode ser aceitável caso o requisito de fato entregue algo com mais de um propósito. Quando uma coluna possui mais de uma associação, isto pode indicar, também, redundância dos requisitos ou excesso de abstração. Pode, também, indicar um desnível natural de abstração entre necessidades (que são mais abstratas) e os requisitos (que são naturalmente mais detalhados).

5.10 Discussão sobre o Caso de Estudo

O caso de estudo apresentado neste capítulo demonstrou a aplicação da técnica proposta em uma situação real de especificação de requisitos de uma empresa que se dispôs a ser voluntária neste estudo. Entretanto, este estudo foi parcial, ou seja, abordou-se apenas uma parte da solução de *marketplace* que a empresa intenciona desenvolver em razão do estágio ainda prematuro do desenvolvimento na época da realização deste estudo. A solução completa envolve um número maior de *stakeholders*, principalmente externos (incluindo compradores, produtores e afiliados) e, consequentemente, levaria a outras demandas, necessidades e requisitos, além daqueles considerados no caso de estudo. Ainda assim, o estudo tratou de questões (*i.e.*, problemas e necessidades) realísticas da empresa.

Com base na conceituação de *stakeholder*, apresentada na subseção 3.4, determinou-se o principal *stakeholders* desta solução e estabeleceu-se informações essenciais, notadamente, os interesses do *stakeholder*. Após identificar os interesses da empresa compradora, a aplicação da abordagem PBSRS, juntamente com a técnica proposta nesta dissertação, ajudou a especialista em requisitos a conduzir as reuniões com o cliente. O objetivo foi organizar as solicitações e as expressões do cliente durante as reuniões de alinhamento em cada etapa da especificação (*i.e.*, problemas, faltas, entregáveis, necessidades e requisitos).

Outro ponto importante a destacar é que a especificação das faltas identificadas pelo *stakeholders*, a partir da análise dos problemas, seguida pela definição dos entregáveis, facilitou a especificação das necessidades e da elaboração dos requisitos. Isso resultou em uma solução mais alinhada com a demanda ou interesse do *stakeholder* e permitiu identificar melhorias em funcionalidades já existentes, conforme os *feedbacks* dos principais clientes da empresa.

6 DISCUSSÃO

Neste capítulo é feita uma discussão sobre as três questões de pesquisa levantadas ao início dos trabalhos e descritas no capítulo de introdução.

6.1 Consistência com Contexto de Especificação de Requisitos

A primeira questão de pesquisa (Q1) diz respeito à consistência do conceito de “Necessidades dos Clientes” dentro do contexto da especificação de requisitos de software e se este conceito pode auxiliar na solução dos problemas P2 e P3, e, consequente, P1. Com a realização desta pesquisa, pode-se considerar que o conceito é, de fato, consistente com a especificação de requisitos de software, pelas seguintes razões:

- Segundo a PBSRS, por definição, todo produto se presta a resolver ou minimizar problemas dos clientes. Assim, a utilidade de um produto, expressa na forma de seus requisitos, deveria estar devidamente alinhada com os problemas a serem resolvidos. Entretanto, associar requisitos a problemas não é fácil nem intuitivo, pois são conceitos cujo relacionamento não é direto e que são definidos em contexto diferentes (*i.e.*, problemas são definidos no contexto do negócio e requisitos são definidos no contexto técnico da solução), além de terem níveis de abstração muito diferentes (*i.e.*, problemas são muito mais abstratos, enquanto requisitos são muito mais detalhados, do ponto de vista técnico). Assim, as “Necessidades dos Clientes” representam uma especificação intermediária entre problemas e requisitos que permite estabelecer uma associação mais precisa e fundamentada entre os requisitos e os problemas que se busca resolver.
- Percebeu-se, por meio do desenvolvimento desta pesquisa, dos exemplos e o caso de estudo, que o conceito de Necessidades dos Clientes representa um conceito natural no raciocínio lógico de análise e concepção de uma solução. Quando o especialista se defronta com um conjunto de problemas dos clientes para o qual pretende propor uma solução, percebe-se ser natural pensar no que esta solução deverá prover, antes de se pensar em como ela será e no que fará. Deve-se considerar, seguramente, que mais de uma ideia ou proposta do que uma solução pode prover pode surgir desta reflexão e corresponderiam a alternativas de solução. Assim, o conceito de

Necessidades dos Clientes demonstrou ser consistente com a cadeia de raciocínio da especificação de requisitos. Entretanto, um estudo maior e mais independente com profissionais de requisitos poderia ser desenvolvido para melhor fundamentar esta observação.

- O conceito de Necessidades dos Clientes pode auxiliar na resolução do problema P1, pois ele permite melhorar o entendimento da utilidade de cada requisito na resolução dos problemas elencados na especificação. Este conceito, também, auxilia na resolução do problema P2, pois permite a análise de completeza e corretude, a partir do estabelecimento das associações entre problemas, necessidades e requisitos. Ainda, o conceito auxilia na resolução do problema P3, pois permite relacionar explicitamente os requisitos com os propósitos (*i.e.*, os problemas a serem resolvidos).

6.2 Contribuição Técnica dos Conceitos de Necessidades

A segunda questão de pesquisa (Q2) diz respeito à contribuição técnica do conceito de Necessidades dos Clientes com a atividade de requisito de software e seu auxílio na solução dos problemas P2 e P3, e, consequente, P1. Com a realização desta pesquisa, pode-se considerar que o conceito traz, de fato, contribuição técnica, pelas seguintes razões:

- A aplicação dos conceitos de Necessidades dos Clientes, apoiada na ontologia e técnica proposta nesta dissertação, fornece uma contribuição técnica ao especialista em requisitos na forma de um ferramental adicional (*i.e.*, além do conhecimento e ferramental clássicos de requisitos de software), visando uma especificação de requisitos de melhor qualidade para uma solução de software.
- Tecnicamente, ou seja, na prática da atividade de requisitos de software, os conceitos de Necessidades dos Clientes, provêm ao especialista em requisitos a capacidade desenvolver as especificações apoiado em associações entre os três domínios de conhecimento (Problemas, Necessidades e Requisitos) que deixam explícitas as coberturas entre eles, ou seja, se as especificações estão completas e corretas. Neste sentido, os conceitos contribuem para resolução de P2.

- A aplicação dos conceitos de Necessidades dos Clientes permite estabelecer tecnicamente (*i.e.*, especificar) as fontes para os requisitos, contribuindo para a resolução do problema P3.
- A aplicação dos conceitos de Necessidades dos Clientes contribui para especificar softwares úteis e viáveis, ou seja, softwares alinhados com os problemas identificados do cliente, melhorando as chances de sucesso do software, contribuindo para a resolução do problema P1.

6.3 Consistência dos Conceitos com Outras Abordagens

Discute-se, nas subseções a seguir, a consistência do conceito de "Necessidades dos Clientes" com outras abordagens de especificação de requisitos.

6.3.1 GORE – Goal Oriented Requirements Engineering

Tanto a PBSRS, quanto a abordagem GORE, estendem a especificação além dos conceitos de requisitos funcionais e não funcionais. Entretanto, as abordagens e os conceitos propostos não são os mesmos, como discutido a seguir.

A abordagem GORE propôs a especificação "orientada a objetivos", ou seja, que a especificação de requisitos parta da definição prévia de "objetivos". Adicionalmente, como apresentado na seção 2.2 deste documento, o GORE, também, faz uso do conceito de "acordo" entre atores para identificar dependências entre eles, como uma forma de modelar uma visão dos negócios e suas dependências em torno de objetivos mais abstratos. Destes acordos, os objetivos são decompostos em requisitos. Assim, os objetivos na abordagem GORE definem uma intenção ou propósito para o software ou sistema a ser desenvolvido e os requisitos associados descrevem as exigências técnicas a serem atendidas pelo futuro produto (em termos funcionais e não funcionais).

Na abordagem PBSRS, tem-se algo análogo em que se propõe dois novos domínios (*i.e.*, Domínio dos Problemas e Domínio das Necessidades), além do Domínio dos Requisitos. O domínio imediatamente associado ao dos requisitos é o das Necessidades dos Clientes. Nele são identificadas as faltas percebidas pelo cliente (*i.e.*, as necessidades) para atendimento de seus problemas. Percebe-se haver uma consistência entre esta visão de "necessidades" da PBSRS e a visão de "objetivos" do GORE. Isso se deve ao fato de que um "objetivo" é uma expressão de

necessidade, pois ele determina algo que o cliente pretende ou procura atingir e que, semanticamente, equivale a uma expressão de falta ou de busca por algo que ele não tem, ou seja, uma necessidade. Na verdade, necessidades podem ser expressas utilizando vários termos equivalentes como necessitar, precisar, desejar, querer, objetivar ou almejar. Todos remetem à mesma ideia de falta de algo que poderia ser suprido pela solução e que permitiria resolver ou minimizar problemas.

A abordagem GORE é orientada exclusivamente ao conceito de “objetivos” o que, na opinião da autora, a torna mais restritiva do que a abordagem PBSRS. Nem sempre refletir e expressar o que falta ao cliente na forma de objetivos é a maneira mais intuitiva para expressar suas faltas. Neste sentido, a abordagem PBSRS é mais flexível e mais abrangente em capturar e expressar as causas dos requisitos. Adicionalmente, a abordagem PBSRS é ainda mais ampla do que o GORE, pois é orientada à resolução de problemas, em vez de focar exclusivamente no alcance dos objetivos que são consequências dos problemas.

O GORE ainda aplica interpretações como a de “Objetivos Funcionais” (*i.e.*, *Functional Goals*) que remetem a coisas que o produto deverá fazer, como por exemplo “*Schedule Meeting*” (*i.e.*, “Agendar uma Reunião”) ou “*Keep doors closed when moving*” (*i.e.*, “Manter a porta fechada durante a movimentação”) (ROLLAND e SALINESI, 2005). Na verdade, esta interpretação pode ser confusa ao especialista em requisitos, pois confunde-se objetivo com requisitos funcionais.

6.3.2 Elementos de Valor (Elements of Value)

O conceito de valor, conforme discutido na seção 2.3, refere-se aos “benefícios” percebidos pelos *stakeholders* em relação ao produto, também, chamado de “entrega de valor” no contexto empresarial, e é usado como estratégia para atração de clientes e para melhorar o seu posicionamento no mercado (conforme discutido na seção 2.3 desta dissertação).

Durante a realização desta pesquisa de mestrado, não se encontrou referências bibliográficas que discutissem a relação entre os Elementos de Valor e requisitos de software e de produtos em geral. A autora realizou alguns experimentos e estudos a respeito, porém eles não foram reportados neste documento porque ficaram inconclusivos. Um fragmento destes estudos é ilustrado a seguir.

Considerou-se o problema apresentado na seção 5.1.3:

P02: A Fantasia Company deve ofertar meios para promover as primeiras vendas dos produtores iniciantes, sob pena de desapontamento e desistência de suas tentativas de tornarem-se produtores de conteúdo e de perda de clientes produtores de conteúdo.

Diante deste problema, a empresa Fantasia Company visualiza uma solução *web* que possa ajudar a resolver ou minimizar esse problema. Esta solução *web* seria um *marketplace*. Com isso, o especialista prospecta a falta de capacidade da empresa para promover as primeiras vendas dos conteúdos digitais dos produtores iniciantes, que poderia ser suprida pelo *marketplace*. Depois, ele identifica que o entregável para suprir esta falta seria o automatismo para promover as primeiras vendas dos conteúdos digitais dos produtores iniciantes e identifica a seguinte necessidade. Por fim, o especialista especificaria os requisitos para atendimento de tal necessidade, conforme recorte abaixo do caso de estudo da seção 5.1.

N04: A Fantasia Company precisa de um *marketplace* para automatizar a promoção das primeiras vendas de conteúdos digitais dos Produtores Iniciantes.

RF10: O *marketplace* deverá destacar os conteúdos de primeira venda em uma prateleira chamada “Première”, enquanto não efetuar as suas três primeiras vendas.

RF11: O *marketplace* deverá gerar promoções com descontos em conteúdos digitais até a realização das dez primeiras vendas.

Neste contexto, pode-se identificar os seguintes elementos de valor associados a esta necessidade.

(i) Motiva para tentativa de carreira de produtor iniciante de conteúdo;
(ii) Informa ao mercado sobre as ofertas dos produtores iniciantes de conteúdo;
(iii) Conecta os produtores iniciantes de conteúdo com clientes interessados;
(iv) Gera renda para as primeiras vendas aos produtores iniciantes;

(v) Organiza as vendas dos produtores de conteúdo, incluindo os iniciantes.

A associação entre as necessidades dos clientes e os elementos de valor parece contribuir para a criação de produtos cujos benefícios poderiam ser melhor percebidos pelos *stakeholder*, atrelados às necessidades e requisitos. Porém, mais estudos seriam necessários para a proposição de conceitos e técnicas a respeito, o que ficou fora do escopo deste trabalho de mestrado.

6.3.3 História de Usuário

As Histórias de Usuários, conforme discutido na seção 2.5, são uma técnica de especificação de requisitos popular, usada em diferentes processos, inclusive em métodos ágeis. O diferencial desta abordagem está na inclusão (*i.e.*, associação) do requisito (*i.e.*, objetivo) com o benefício (*i.e.*, valor de negócio) esperado com ele. Nota-se, portanto, que as Histórias de Usuários estendem a especificação dos requisitos associando-os a algum "benefício", ou seja, a alguma causa que os justificam. Percebe-se, na literatura sobre esta abordagem, uma falta de consistência e de profundidade na caracterização precisa deste conceito de "benefício". Não é incomum, por exemplo, que os benefícios sejam genéricos, mal definidos ou remetendo a resultados e não causas. Por exemplo, Amna e Poels (2022) apresenta a seguinte história de usuário: "As a customer, I want to transfer funds between my linked accounts, so that I can fund my credit card" (*i.e.*, "Como cliente, quero transferir fundos entre minhas contas vinculadas para poder financiar meu cartão de crédito"). Ou seja, a semântica deste conceito não está clara, além do fato de que o conceito de "benefício" também já é empregado na visão Elementos de Valor (conforme discutido na seção 6.3.2) com outra interpretação.

Embora esta pesquisa de mestrado não tenha se aprofundado na abordagem de Histórias de Usuários, pode-se perceber que existe uma boa consistência dela com a abordagem PBSRS. A associação de uma causa aos requisitos proposta nas Histórias de Usuário é análoga a associação entre as necessidades do cliente e os requisitos de software proposta na abordagem PBSRS. Em ambas, os requisitos se associam a(s) causa(s) de sua especificação, permitindo inclusive checar se eles estão corretos e completos.

A abordagem PBSRS, entretanto, é mais ampla do que a de Histórias de Usuários, pois alcança a especificação dos problemas, além da camada de

associação de necessidades do cliente. Ela é, também, mais abrangente, dado que o conceito de necessidades é mais genérico e compreensível do que o de benefícios que é mais limitante. A abordagem PBSRS é, ainda, mais sistemática na sua aplicação, levando, por exemplo, à possibilidade de análises de cobertura e grau de atendimento dos requisitos.

7 CONCLUSÃO

Esta pesquisa teve como tema a Especificação das Necessidades dos Clientes e seu relacionamento com os requisitos de software que é uma das etapas dentro da abordagem de especificação denominada *Problem Bases Software Requirements Specification* (PBSRS). Como principais contribuições, esta pesquisa apresenta um aprofundamento dos conceitos; a proposta de uma ontologia para especificação de requisitos de software; a proposição de uma técnica para especificação e relacionamento entre as necessidades dos clientes e os requisitos de software; e um caso de estudo aplicando a técnica proposta.

O aprofundamento dos conceitos de Necessidades do Cliente contribui para um melhor entendimento e melhor diferenciação do termo Necessidade, estendendo os estudos iniciais sobre PBSRS apresentados por Ana Pereira (Pereira, 2011) e Rafael Gorski de Souza (Souza, 2016).

Visando tornar mais precisos os conceitos e relações empregados na abordagem PBSRS, esta pesquisa desenvolveu uma ontologia de requisitos. Esta ontologia incluiu conceitos de *stakeholders*, interesse, *glance*, *vision*, problemas, necessidades e requisitos e suas relações.

A pesquisa propôs uma técnica para a Especificação de Necessidades dos Clientes. Esta técnica se baseia na análise e especificação de faltas e entregáveis para definição da causa ou justificativa (*i.e.*, necessidades dos clientes). Desta forma, espera-se reduzir o distanciamento entre os requisitos e as necessidades do cliente que, consequentemente, poderá aumentar a chance de sucesso dos desenvolvimentos de software no sentido de melhor atender aos interesses dos clientes pelo software.

Um caso de estudo foi desenvolvido com o objetivo de avaliar os conceitos e a técnica propostos. Este estudo permitiu observar as qualidades da proposta, notadamente, na contribuição que traz na precisão e clareza das especificações, a partir de uma situação parcial e real de sistema de informação em uma empresa voluntária. A abordagem ajudou a autora a identificar, de forma efetiva, os interesses e problemas da empresa, facilitando o aprofundamento na análise dos problemas, como a ideação e o sentimento de falta. Com este entendimento, a especificação dos entregáveis, necessidades e requisitos tornaram-se menos complexos,

resultando em uma solução mais alinhada aos interesses e problemas dos *stakeholders*.

A pesquisa também abordou três questões centrais: (Q1) a consistência do conceito de “Necessidades do Cliente” na especificação de requisitos de software, (Q2) sua contribuição técnica para a atividade de Requisitos de Software e (Q3) sua consistência com abordagens GORE, Elementos de Valor e Histórias de Usuário. A resposta a essas questões mostrou que o conceito de Necessidades dos Clientes, junto com a ontologia e a técnica propostos nesta dissertação, contribui para resolver os problemas: (P2) da dificuldade do especialista em requisitos em definir um conjunto de requisitos completos e corretos; (P3) da dificuldade de compreender as reais demandas dos clientes, uma vez que as informações disponíveis são frequentemente vagas e incompletas; e, por consequência, mitiga o risco de insucesso dos produtos de software (P1).

Em síntese, esta dissertação apresenta uma contribuição à abordagem PBSRS com a proposta de uma ontologia para a atividade de Requisitos de Software junto com a proposta de uma técnica de Especificação de Necessidade. Esta pesquisa visou auxiliar o especialista em requisitos a reduzir um problema gravíssimo que é a capacidade de construção de um software que seja considerado útil ou viável pelo seu mercado comprador. Ao abordar este ponto crítico, a proposta desta dissertação busca aumentar a precisão da identificação de necessidades e, consequentemente, elevar a taxa de sucesso dos projetos de software.

A pesquisa também abriu caminhos para trabalhos futuros, como:

- Desenvolver novos experimentos reais com profissionais da área visando colher suas impressões, críticas e sugestões sobre a PBSRS;
- Aprofundar os estudos sobe Elementos de Valor e sua associação com a atividade de Requisitos de Software;
- Aprofundar a classificação de necessidades e incluí-la na ontologia para a atividade de Requisitos de Software;
- Explorar a aplicação da teoria de Projeto Axiomático na especificação de problemas e necessidades dos clientes e
- Especificar e desenvolver uma ferramenta computacional de apoio à PBSRS.

REFERÊNCIAS

ABD ELWAHAB, Khloud; LATIF, Mahmoud Abd EL; KHOLEIF, Sherif. **Identify and Manage The Software Requirements Volatility.** International Journal of Advanced Computer Science and Applications, v. 7, n. 5, 2016.

ABDALAZEIM, Alaa; MEZIANE, Farid. **A review of the generation of requirements specification in natural language using objects UML models and domain ontology.** Procedia Computer Science, v. 189, p. 328-334, 2021.

ALJAHDALI, Sultan; BANO, Jameela; HUNDEWALE, Nisar. **Goal oriented requirements engineering-a review.** In: 24th International Conference on Computer Applications in Industry and Engineering, Honolulu, Hawaii, USA, CAINE. 2011. p. 16-18.

ALMQUIST, Eric; SENIOR, John; BLOCH, Nicolas. **The elements of value.** Harvard business review, v. 94, n. 9, p. 47-53, 2016.

ALVES, Vander et al. **Requirements engineering for software product lines: A systematic literature review.** Information and Software Technology, v. 52, n. 8, p. 806-820, 2010.

AMNA, Anis R.; POELS, Geert. **Systematic literature mapping of user story research.** IEEE Access, v. 10, p. 51723-51746, 2022.

ASSYNE, Nana; GHANBARI, Hadi; PULKKINEN, Mirja. **The state of research on software engineering competencies: A systematic mapping study.** Journal of Systems and Software, v. 185, p. 111183, 2022.

BECK, Kent. **Embracing change with extreme programming.** Computer, v. 32, n. 10, p. 70-77, 1999.

BAGHERI, Samaneh et al. **A reference model-based user requirements elicitation process: Toward operational business-IT alignment in a co-creation value network.** Information and Software Technology, v. 111, p. 72-85, 2019.

BAYUS, Barry L.; SHANE, S. **Understanding customer needs.** Handbook of Technology and Innovation Management, p. 115-142, 2008.

BOURQUE P.; FAIRLEY R. E. **Guide to the Software Engineering Body of Knowledge**, Version 3.0, IEEE Computer Society, 2014. Available in: https://www.computer.org/education/bodies-of-knowledge/software-engineering. Acessado em 20/03/202.

CAMBRIDGE DICTIONARY. Available in: https://dictionary.cambridge.org/dictionary/english-portuguese/need. Acessado em 01/12/2022.

CICO, Orges et al. **Exploring the intersection between software industry and Software Engineering education-A systematic mapping of Software Engineering Trends.** Journal of Systems and Software, v. 172, p. 110736, 2021.

COHN, Mike. **User stories applied: For agile software development.** Addison-Wesley Professional, 2004.

DALTON, Jeff; DALTON, Jeff. **Acceptance Testing.** Great Big Agile: An OS for Agile Leaders, p. 111-112, 2019.

FERNÁNDEZ, D. Méndez et al. **Naming the pain in requirements engineering: Contemporary problems, causes, and effects in practice.** Empirical software engineering, v. 22, p. 2298-2338, 2017.

GERACIE, Greg; EPPINGER, Steven D. (Ed.). **The Guide to the Product Management and Marketing Body of Knowledge.** Product Management Educational Institute (PMEI), 2013.

GIORGINI, Paolo; MYLOPOULOS, John; SEBASTIANI, Roberto. **Goal-oriented requirements analysis and reasoning in the tropos methodology.** Engineering Applications of Artificial Intelligence, v. 18, n. 2, p. 159-171, 2005.

GIRAY, Görkem. **A software engineering perspective on engineering machine learning systems: State of the art and challenges.** Journal of Systems and Software, v. 180, p. 111031, 2021.

GRIGG, Nigel P. **Redefining quality in terms of value, risk and cost: a literature review.** International Journal of Quality & Reliability Management, v. 38, n. 5, p. 1065-1089, 2021.

GROEN, Eduard C. et al. **The crowd in requirements engineering: The landscape and challenges.** IEEE software, v. 34, n. 2, p. 44-52, 2017.

GUARINO, Nicola; OBERLE, Daniel; STAAB, Steffen. **What is an ontology?.** Handbook on ontologies, p. 1-17, 2009.

HAPPEL, Hans-Jörg; SEEDORF, Stefan. **Applications of ontologies in software engineering.** In: Proc. of Workshop on Sematic Web Enabled Software Engineering"(SWESE) on the ISWC. 2006. p. 5-9.

HORKOFF, Jennifer et al. **Goal-oriented requirements engineering: an extended systematic mapping study.** Requirements engineering, v. 24, p. 133-160, 2019.

HORKOFF, Jennifer; YU, Eric. **Interactive goal model analysis for early requirements engineering.** Requirements Engineering, v. 21, p. 29-61, 2016.

IEEE COMPUTER SOCIETY. SOFTWARE ENGINEERING STANDARDS COMMITTEE; IEEE-SA STANDARDS BOARD. **IEEE recommended practice for software requirements specifications**. IEEE, 1998.

INAYAT, Irum et al. **A systematic literature review on agile requirements engineering practices and challenges.** Computers in human behavior, v. 51, p. 915-929, 2015.

INCOSE, **System Engineering Handbook, A Guide for System Life Cycle Processes and Activities**, Hoboken, Wiley, 2015.

ISOTANI, Seiji; BITTENCOURT, Ig Ibert. **Dados abertos conectados: em busca da web do conhecimento.** Novatec Editora, 2015.

KAIYA, Haruhiko; SAEKI, Motoshi. **Using domain ontology as domain knowledge for requirements elicitation.** In: 14th IEEE International Requirements Engineering Conference (RE'06). IEEE, 2006. p. 189-198.

KAMALRUDIN, Massila; SIDEK, Safiah. **A review on software requirements validation and consistency management.** International journal of software engineering and its applications, v. 9, n. 10, p. 39-58, 2015.

KROLL, Robert J. **Advancing the consumer interest: traditional ideology versus a generic view.** Advancing the Consumer Interest, p. 22-27, 1991.

LAPLANTE, Phillip A.; KASSAB, Mohamad. **Requirements engineering for software and systems.** Auerbach Publications, 2022.

LI, Jinyu et al. **Attributes-based decision making for selection of requirement elicitation techniques using the analytic network process.** Mathematical Problems in Engineering, v. 2020, n. 1, p. 2156023, 2020.

LIM, Sachiko; HENRIKSSON, Aron; ZDRAVKOVIC, Jelena. **Data-driven requirements elicitation: A systematic literature review.** SN Computer Science, v. 2, n. 1, p. 16, 2021.

LUCASSEN, Garm et al. **The use and effectiveness of user stories in practice.** In: Requirements Engineering: Foundation for Software Quality: 22nd International Working Conference, REFSQ 2016, Gothenburg, Sweden, March 14-17, 2016, Proceedings 22. Springer International Publishing, 2016. p. 205-222.

MAVIN, Alistair et al. **Does goal-oriented requirements engineering achieve its goal?.** In: 2017 IEEE 25th international requirements engineering conference (RE). IEEE, 2017. p. 174-183.

MICHAELIS, **Melhoramentos**. Available in: https://michaelis.uol.com.br/moderno-portugues/busca/portugues-brasileiro/necessidade/. Acessado em 01/12/2023.

MIKULIĆ, Josip; PREBEŽAC, Darko. **A critical review of techniques for classifying quality attributes in the Kano model**. Managing Service Quality: An International Journal, v. 21, n. 1, p. 46-66, 2011.

MORANDINI, Mirko et al. **Engineering requirements for adaptive systems.** Requirements Engineering, v. 22, n. 1, p. 77-103, 2017.

NAUR, P., RANDELL, B. **Software Engineering**, Report on a conference sponsored by the NATO SCIENCE COMMITTEE Garmisch, Germany, October 1968, 1969.

NAVARRO-ALMANZA, Raul; JUAREZ-RAMIREZ, Reyes; LICEA, Guillermo. **Towards supporting software engineering using deep learning: A case of software requirements classification.** In: 2017 5th International Conference in Software Engineering Research and Innovation (CONISOFT). IEEE, 2017. p. 116-120.

OKESOLA, Olatunji J. et al. **Qualitative comparisons of elicitation techniques in requirement engineering.** ARPN J. Eng. Appl. Sci, v. 14, n. 2, p. 565-570, 2019.

PEREIRA, A. M. **Abordagem de especificação de requisitos baseada em projeto axiomático**. Dissertação de Mestrado, Universidade Tecnológica Federal do Paraná, Curitiba, p. 168, 2011.

ROLLAND, Colette; SALINESI, Camille. **Modeling goals and reasoning with them.** Engineering and managing software requirements, p. 189-217, 2005.

SCHÖN, Eva-Maria; THOMASCHEWSKI, Jörg; ESCALONA, María José. **Agile Requirements Engineering: A systematic literature review.** Computer standards & interfaces, v. 49, p. 79-91, 2017.

SEBOK EDITORIAL BOARD. **The Guide to the Systems Engineering Body of Knowledge** (SEBoK), v. 2.8, R.J. Cloutier (Editor in Chief). Hoboken, NJ: The Trustees of the Stevens Institute of Technology, 2023.

SILVIA, Paul J. **Interest—The curious emotion.** Current directions in psychological science, v. 17, n. 1, p. 57-60, 2008.

SINAGA, Sarman; LUMBAN GAOL, Jonner; ICHSAN, Reza Nurul. **The effect of product innovation on consumer interest in the purchase of bottled tes product at PT.** Sinar sosro medan. Budapest International Research and Critics Institute (BIRCI-Journal): Humanities and Social Sciences, v. 4, n. 1, 2021.

SINGH, Prateek; SINGH, Deepali; SHARMA, Ashish. **Rule-based system for automated classification of non-functional requirements from requirement specifications.** In: 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI). IEEE, 2016. p. 620-626.

PEREIRA, A. M. **Abordagem de Especificação de Requisitos Baseada em Projeto Axiomático**. Dissertação de Mestrado, Universidade Tecnológica do Paraná, Curitiba, p. 168, 2016

SOUZA, R. G. M. **Problem-Based SRS: Método para Especificação de Requisitos de Software Baseado em Problemas**. Dissertação de Mestrado, Universidade Tecnológica Federal do Paraná, Curitiba, p. 126, 2016.

SOUZA, Rafael Gorski M.; STADZISZ, Paulo Cézar. **PROBLEM-BASED SOFTWARE REQUIREMENTS SPECIFICATION.** Revista Eletrônica de Sistemas de Informação, v. 15, n. 2, 2016.

SUÁREZ-FIGUEROA, Mari Carmen; GÓMEZ-PÉREZ, Asunción. **Ontology requirements specification.** In: Ontology Engineering in a Networked World. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. p. 93-106.

SUH, N. P. **Complexity: Theory and Applications**. Oxford University Press, New York, 2005

SUH, NAM P. **The Principles of Design.** Oxford University Press, 1990.

TENBERGEN, Bastian; DAUN, Marian. **Is requirements-engineering research delivering what it promised?: a review of its accomplishments and opportunities after 10 years.** IEEE Software, v. 36, n. 4, p. 6-11, 2019.

THACKER, Ben H. et al. **Concepts of model verification and validation.** 2004.

THE STANDISH GROUP INTERNATIONAL, Inc., **The Chaos Report**, 1995.

TRKMAN, Marina et al. **Impact of the conceptual model's representation format on identifying and understanding user stories.** Information and software technology, v. 116, p. 106169, 2019.

VAN LAMSWEERDE, Axel. **Goal-oriented requirements enginering: a roundtrip from research to practice [enginering read engineering].** In: Proceedings. 12th IEEE International Requirements Engineering Conference, 2004. IEEE, 2004. p. 4-7.

VAN HARMELEN, Frank; LIFSCHITZ, Vladimir; PORTER, Bruce (Ed.). **Handbook of knowledge representation.** Elsevier, 2008.

WANG, Tianci et al. **A Deep Learning-Based Method for Identifying User Story Semantic Conflicts.** In: 2023 IEEE 23rd International Conference on Software Quality, Reliability, and Security Companion (QRS-C). IEEE, 2023. p. 220-229.

ZONG-YONG, L. I. et al. **Towards a multiple ontology framework for requirements elicitation and reuse.** In: 31st Annual International Computer Software and Applications Conference (COMPSAC 2007). IEEE, 2007. p. 189-195.

ANEXO A - Tesauro

TESAURO

Conceito	Definição	Exemplo
Cliente	É aquele ou aquilo que irá adquirir e utilizar o produto a ser desenvolvido.	Consumidor, comprador, adquiridor ou usuário.
Cliente Institucional	É um segmento de uma instituição específica que irá adquirir e utilizar o produto a ser desenvolvido.	Indústrias de laticínios.
Cliente Mercado Institucional	É um segmento do mercado (público-alvo) que irá adquirir e utilizar o produto a ser desenvolvido.	Mercado de Instituições Financeiras
Cliente Mercado Pessoal	É uma parte do mercado de pessoas físicas que poderá adquirir e utilizar o produto.	Estudantes universitários.
Cliente Setor Institucional	É um segmento de um determinado setor dentro de uma instituição específica que irá adquirir e utilizar o produto a ser desenvolvido, atendendo particularmente às suas necessidades.	Diretor e os funcionários do setor de estoque de uma fábrica de automóveis.
Condição	É a informação de quando se aplica o requisito.	O software deverá ativar o alarme de incêndio em no máximo 2 segundos, quando receber o sinal do sensor.
Controle	É algo que os softwares podem prover envolvendo o monitoramento e ação contínua, seja sobre dados ou sobre ambientes físicos.	Automatismo de regulagem de climatização de um ambiente.
Criação de Objeto	É a capacidade dos softwares em prover meios para criar objetos digitais na forma de texto, gráficos, modelos, tabelas, apresentações, fórmulas, vídeos, entre outros.	Bloco de notas, *dashboards* ou Excel.
Entretenimento	É software que pode fornecer	Jogos, reprodutores de música

	diversão, distração, lazer, recreação ou passatempos.	e vídeo, leitores de texto ou mídias sociais.
Esperança	É algo que se imagina a ser possível e que o *stakeholder* poderia fazer ou prover, que poderia ser entendido como um diferencial ou alguma coisa a mais que é ofertada.	Os clientes de uma empresa de telecomunicações têm a esperança de receber uma comunicação de felicitações pelo seu aniversário.
Expectativa	É quando alguém ou alguma coisa espera algo do *stakeholder*, no sentido de considerar que esse algo é factível, possível e desejado, mesmo que não seja obrigatório.	Os clientes de um fabricante de automóveis podem ter a expectativa de que novos modelos serão melhores em termos de redução de emissões, além das exigências regulatórias, caso contrário, poderá haver uma redução de atratividade sobre os automóveis ofertados.
Fonte	É meio, sistema ou produto que proveria as carências do *stakeholder*.	Meio, sistema ou produto. O fazendeiro necessidade de um software para ter conhecimento da temperatura ambiente.
Funcionalidade	Representa uma parte das ações que um software deverá realizar.	Gerar um alarme.
Glance	É uma especificação abstrata da solução visando resolver um problema do cliente e que, assim, sabe-se qual é o meio ou o tipo de produto almejado.	Conjunto de uma aplicação *web* com um *back-end* e banco de dados destinado a atender usuários para provimento de informações sobre o trânsito da cidade.
Importância	É o julgamento da relevância ou impacto feita pelo *stakeholder*, levando em conta a potencial utilidade ou as contribuições do produto para os negócios do *stakeholder*.	Alta, média ou baixa importância.
Informação	Conteúdos que auxiliam na construção de conhecimento	Notícias, relatórios, *dashboards*, estatísticas, tabela de dados ou

	para os usuários.	textos.
Intensidade	É a medida ou grau da punição.	Obrigação, expectativa ou esperança.
Interesse	O “Interesse” do *stakeholder* está relacionado à sua percepção de relevância, importância, vantagem ou utilidade em relação ao “Objeto do Interesse”.	Realizar transição de carreira para produtor digital.
Necessidade	É “o que se precisa” do ponto de vista do que o software deverá “entregar” ou “prover”.	Ter conhecimento do vencimento do processo.
Nome	Nome de identificação do *stakeholder*.	Produtores iniciantes de conteúdo.
Objeto da Necessidade	É a coisa que o *stakeholder* quer que seja provida, ou seja, que possui carência.	Dar alta visibilidade a sua oferta visando a primeira venda.
Objeto de Interesse	Remete a algo que o *stakeholder* considera útil.	Realizar transição de carreira para produtor digital.
Objeto do Problema	É algo que descreve a dificuldade que é a fonte do problema, ou seja, a obrigação à qual o *stakeholder* está submetido.	Realizar a primeira venda.
Objeto do Requisito	É algo que sofre a ação do “Verbo de Ação”, ou seja, o objeto da ação.	O software deverá registrar a ocorrência de um alarme, o SGP deverá gerar uma notificação ao usuário e o robô Laura deverá identificar uma infecção.
Obrigação	É um dever que recai sobre o *stakeholder* devido à imposição de uma penalidade.	Declaração anual e ajustes de imposto de renda.
Parte da Fonte	É um componente da solução ao qual um requisito se aplica.	O módulo de *login* deverá checar o *id* e senha digitados.
Penalização	É a consequência do descumprimento da obrigação.	Sinalização, dor ou prejuízo.
Problema	É a obrigação que recai sobre um *stakeholder* em relação ao seu negócio.	Produtores iniciantes de conteúdo têm obrigação de realizar a primeira venda sob

		pena de desapontamento e desistência de sua tentativa de carreira como produtores de conteúdo digital.
Requisito	É algo que é exigido pelo *stakeholder*, no sentido do que o futuro software deverá fazer ou ter como característica para que uma ou mais necessidades sejam atendidas.	O software deverá gerar um alarme de incêndio.
Requisito Funcional	É a função que o software deve executar.	Formatar um texto ou modular um sinal.
Requisito Não Funcional	É aquele que atua para restringir a solução. Eles podem ser classificados em requisitos de desempenho, manutenção, segurança, interoperabilidade entre outros tipos de requisitos de software.	O software deverá ser desenvolvido em linguagem, C++.
Restrições	É a propriedade ou característica, em geral limitantes, à ação prevista no requisito.	O software deverá ativar o alarme de incêndio em no máximo 2 segundos.
Stakeholder	Aquele (*i.e.*, um indivíduo ou coletivo de indivíduo) ou aquilo (*i.e.*, uma organização) que possui interesse na solução.	Cliente, consumidor, usuário ou público.
Stakeholder Externo	Aquele ou aquilo que não está inserido na rotina da empresa proponente do software, mas que possui interesse na solução.	Fornecedores, governos, Organizações Não Governamentais e associações.
Stakeholder Interno	É aquilo ou aquilo que se encontra dentro do ambiente da empresa proponente do software e que possui interesse na solução.	Acionistas, colaboradores da empresa, setores da empresa, responsáveis, diretores e desenvolvedores.
Sujeito	Designa a solução ou uma parte ou componentes desta que é alvo da exigência expressa pelo	*Marketplace*, software *web* e modulo e *login*.

	requisito.	
Tipo de Necessidade	É o que a fonte poderá prover, quando a fonte for igual a software.	Criação de objetos digitais, controle, entretenimento e informações.
Verbo de Ação	É a exigência imposta pelo requisito e, também, a ação ou função exigida do sujeito do requisito. Em geral, é um verbo conjugado no infinitivo para remeter a ação de ideia efetivamente.	Deverá registrar, deverá gerar, deverá identificar.
Verbo de Propriedade	É a existência de restrições expressas pelos requisitos (ser, ter e conter).	O software deverá ser desenvolvido em linguagem, C++ ou o software deverá ter IHC em tons de cinza.
Vision	É um documento que detalha a solução visando definir o seu escopo, ou seja, uma descrição mais detalhada do conceito de "*Glance*".	Solução *web* de vendas.