UNIVERSIDADE FEDERAL DO PAMPA

Giovane D'Avila Mendonça

Uma Técnica de Inspeção para Documentos de Especificação de Requisitos para Sistemas Multiagentes

Alegrete
2021

Giovane D'Avila Mendonça

Uma Técnica de Inspeção para Documentos de Especificação de Requisitos para Sistemas Multiagentes

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Engenharia de Software da Universidade Federal do Pampa, como requisito parcial para a obtenção do título de Mestre em Engenharia de Software.

Orientador: Prof. Dr. Gilleanes Thorwald Araujo Guedes

Alegrete
2021

Ficha catalográfica elaborada automaticamente com os dados fornecidos
pelo(a) autor(a) através do Módulo de Biblioteca do
Sistema GURI (Gestão Unificada de Recursos Institucionais) .

M539t Mendonça, Giovane D'Avila
Uma Técnica de Inspeção para Documentos de Especificação de Requisitos para Sistemas Multiagentes / Giovane D'Avila Mendonça.
181 p.

Dissertação(Mestrado)-- Universidade Federal do Pampa, MESTRADO EM ENGENHARIA DE SOFTWARE, 2021.
"Orientação: Gilleanes Thorwald Araujo Guedes".

1. Engenharia de Requisitos. 2. Especificação de Requisitos. 3. Validação de Requisitos. 4. Padrão ISO/IEC/IEEE 29148:2018. 5. Sistemas Multiagentes. I. Título.

Giovane D´Avila Mendonça

Uma Técnica de Inspeção para Documentos de Especificação de Requisitos para Sistemas Multiagentes

Dissertação/Tese apresentada ao Programa de Pós-Graduação em Engenharia de Software da Universidade Federal do Pampa, como requisito parcial para obtenção do Título de Mestre em Engenharia de Software.

Dissertação defendida e aprovada em: 29 de setembro de 2021.

Banca examinadora:

_____

Prof. Dr. Gilleanes Thorwald Araujo Guedes

Orientador

UniPampa

_____

Profa. Dra. Rosa Maria Vicari

UFRGS

_____

Prof. Dr. João Pablo Silva da Silva

UniPampa

Assinado eletronicamente por **ROSA MARIA VICARI**, **Usuário Externo**, em 29/09/2021, às 17:37, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **GILLEANES THORWALD ARAUJO GUEDES**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 29/09/2021, às 17:39, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **JOAO PABLO SILVA DA SILVA**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 29/09/2021, às 17:40, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

A autenticidade deste documento pode ser conferida no site https://sei.unipampa.edu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **0623752** e o código CRC **37740416**.

Este trabalho é dedicado à todos(as) que
contribuíram de alguma forma para o seu desenvolvimento.

AGRADECIMENTOS

Agradeço primeiramente à minha família. Minha esposa Denise, filhas Hanna e Valentina que estão sempre me apoiando.

Ao meu orientador Prof. Dr. Gilleanes Thorwald Araujo Guedes, por ter aceitado me orientar. Agradeço por sua dedicação em realizar reuniões mesmo em feriados e fins de semana. Sem sua orientação não seria possível finalizar este trabalho.

Agradeço também aos colegas do grupo de pesquisa Iderli Pereira de Souza Filho e Willian Samuel Gerstberger, que contribuiram no desenvolvimento do processo que estamos propondo.

Por fim, agradeço a Universidade Federal do Pampa (UNIPAMPA) pela oportunidade de um ensino de qualidade de forma gratuita.

Obrigado a todos(as)!

“O fracasso e a invenção são gêmeos inseparáveis.
Para inventar, é preciso experimentar; se você soubesse
de antemão que algo vai dar certo, não seria um experimento”.
(Jeff Bezos)

RESUMO

Sistemas multiagentes são caracterizados por serem compostos por diversos agentes que interagem entre si. Um agente é um processo capaz de realizar ações sem intervenção do usuário, além disso, tem a capacidade de perceber e responder a mudanças no ambiente. No entanto, o desenvolvimento deste tipo de sistema trouxe novos desafios para a engenharia de software. Entre eles, a necessidade de adaptar a Engenharia de Requisitos para o contexto de sistemas multiagentes. A engenharia de requisitos é uma importante área da Engenharia de Software que se preocupa em elicitar, analisar, especificar e validar os requisitos do software para garantir a correta compreensão do que precisa ser desenvolvido. O objetivo da especificação de requisitos é fornecer uma descrição detalhada do que o sistema deve fazer. Ela envolve a produção de um documento que pode ser sistematicamente revisado, avaliado e aprovado. Problemas na especificação de requisitos são apontados como as principais causas de falhas em projetos de software, nesse sentido, a verificação dos requisitos visa garantir a qualidade do software que está sendo desenvolvido. Assim, diversas técnicas de inspeção foram propostas para a verificação de requisitos. Dentro do contexto de inspeções está a técnica de Leitura Baseada em Perspectiva ou *Perspective-Based Reading* (PBR) que apresentou comprovada eficácia na detecção de falhas em requisitos de software. Desse modo, acreditamos que a engenharia de requisitos para Sistemas Multiagentes pode se beneficiar da aplicação dessa técnica, de forma a melhorar e garantir a qualidade da especificação de requisitos. No entanto, a técnica de Leitura Baseada em Perspectiva não permite a inspeção de características particulares de Sistemas Multiagentes. Dessa forma, esta pesquisa tem como objetivo adaptar esta técnica para permitir a verificação de documentos de especificação de requisitos para Sistemas Multiagentes. Também produzimos um formato de documento de especificação de requisitos que suporte o modelo *Belief-Desire-Intention*. Para isso, estendemos o padrão ISO/IEC/IEEE 29148:2018, considerando que, selecionar um padrão é um passo importante para escrever especificações de requisitos. Nossa adaptação da técnica PBR foi desenvolvida especificamente para este modelo de representação de requisitos, todavia acreditamos que ela pode ser aplicada a outros documentos de especificação de requisitos com poucas alterações. É importante destacar que esta técnica de inspeção e a extensão do padrão estão sendo propostas para utilização em um processo específico de Engenharia de Requisitos para Sistemas Multiagentes atualmente em desenvolvimento.

Palavras-chave: Engenharia de Requisitos. Especificação de Requisitos. Validação de Requisitos. Padrão ISO/IEC/IEEE 29148:2018. Sistemas Multiagentes. Modelo BDI.

ABSTRACT

Multiagent systems are characterized by being composed of several agents interacting with each other. An agent is a process capable of performing actions without user intervention, moreover, an agent has the hability to perceive and respond to changes in the environment. However, the development of this kind of system brought new challenges to the software engineering. Among them, the need to adapt the requirements engineering to the context of multi-agent systems. Requirements Engineering is an important area of Software Engineering that is concerned with eliciting, analyzing, specifying, and validating software requirements to ensure the correct understanding of what needs to be developed. The objective of requirements specification is to provide a detailed description of what the system must do. It involves the production of a document that can be systematically reviewed, evaluated, and approved. Problems in the requirements specification are pointed out as the main causes of failures in software projects, in this sense, requirements verification aims to ensure the quality of the software being developed. Thus, several inspection techniques were proposed for requirements verification. Within the context of inspections is the technique of Perspective-Based Reading (PBR) which has been shown to be effective in detecting failures in software requirements. In this way, we believe that requirements engineering for multiagent systems can benefit from the application of this technique, in order to improve and ensure the requirements specification quality. However, Perspective-Based Reading does not allow inspecting specific features of multiagent systems. Thus, this research has as its objective to adapt this inspection technique in order to verify requirements specification documents for multiagent systems. We have also produced a template for requirements specification document that supports the Belief-Desire-Intention model. For this, we extended the ISO/IEC/IEEE 29148:2018 standard, considering that, selecting a documentation standard is an important step for describing requirements specifications. Our adaptation of the PBR technique was specifically developed for this requirements representation template, nevertheless we believe it can be applied to other requirement specification documents with little modifications. It is important to highlight that this inspection technique and the template standard extension are being proposed to be used in a specific requirements engineering process for multiagent systems currently under development.

Key-words: Requirements Engineering. Requirements Specification. Requirements Validation. Multi-agent Systems. BDI Model. Standard ISO/IEC/IEEE 29148:2018.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

AOSE Engenharia de Software Orientada a Agente ou *Agent-Oriented Software Engineering*

BDI crença–desejo–intenção ou *Belief-Desire-Intention*

CQ Criterio de Qualidade

DERS Documento de Especificação de Requisitos de Software

ER Engenharia de requisitos

ERS Especificação de Requisitos de Software

GIRSMA Guia de Inspeção de Requisitos para Sistemas Multiagentes

ICM Incompleto

ICR Incorreto

ICS Inconsistente

IUC *Internal Use Case* ou Caso de Uso Interno

NRA Não rastreável

PBR Leitura Baseada em Perspectiva ou *Perspective-Based Reading*

QP Questão de Pesquisa

RSL Revisão Sistemática da Literatura

SMA Sistema Multiagentes

SWEBOK *Engineering Body of Knowledge* ou Guia do Conhecimento da Engenharia de Software

UML *Unified Modelin Language* ou Linguagem de Modelagem Unificada

SUMÁRIO

1 INTRODUÇÃO

Um agente é um processo autônomo, com habilidade social, reativo e proativo (VICARI; GLUZ, 2007), dessa forma, um agente é capaz de realizar ações sem intervenção do usuário (WOOLDRIDGE; JENNINGS, 1995; PORNPHOL; CHITTAYASOTHORN, 2006; SIVAKUMAR; VIVEKAN; KANIMOZHI, 2013). Ainda, um agente é perceptivo, sendo capaz de perceber e responder a mudanças em seu ambiente (SIVAKUMAR; VIVEKAN; KANIMOZHI, 2013) e agentes podem cooperar entre si para alcançar seus objetivos (DELOACH; WOOD, 2000; WANG; JIANG, 2015). Sistemas Multiagentes são caracterizados por serem compostos por diversos agentes que interagem entre si (WOOLDRIDGE, 2009).

De acordo com Jennings (1999), ao adotar uma visão de mundo orientada a agente, logo se torna evidente que a maioria dos problemas exigem ou envolvem múltiplos agentes, devido a representar múltiplas perspectivas, por ser definida uma natureza descentralizada do problema ou existirem múltiplas áreas de atuação no sistema.

Dessa forma, sistemas multiagentes se tornaram uma alternativa para o desenvolvimento de sistemas complexos (LABBA; SAOUD; DUGDALE, 2015). Todavia, o desenvolvimento desse tipo de sistema trouxe vários desafios para a engenharia de software. Sendo assim, surgiu uma nova área que mescla características de diferentes disciplinas tanto da Engenharia de Software quanto da Inteligência Artificial (CUESTA; GÓMEZ; GONZÁLEZ, 2007), denominada Engenharia de Software Orientada a Agente ou *Agent-Oriented Software Engineering* (AOSE). Segundo Guedes e Vicari (2010), a área de AOSE tem como objetivo adaptar as práticas de engenharia de software especificamente para o desenvolvimento de sistemas baseados em agentes.

Os objetivos da AOSE incluem produzir metodologias, processos, técnicas, linguagens de modelagem e ferramentas para o desenvolvimento de sistemas multiagentes (GENZA; MIGHELE, 2013; MENDONÇA; SOUZA FILHO; GUEDES, 2021). Esses objetivos, implicam na construção de novas abordagens de verificação e validação, diferentes da engenharia de software convencional (FUENTES-FERNÁNDEZ; GÓMEZ-SANZ; PAVÓN, 2004).

Como não poderia deixar de ser, a AOSE inclui a adaptação da Engenharia de Requisitos para Sistemas Multiagentes. A Engenharia de Requisitos é uma área da Engenharia de Software que se preocupa em levantar, analisar, especificar e validar os requisitos do software para garantir a compreensão correta do que precisa ser desenvolvido (FUENTES-FERNÁNDEZ; GÓMEZ-SANZ; PAVÓN, 2009). A engenharia de requisitos é uma etapa crucial na engenharia de software em que determina se um projeto será bem sucedido ou resultará em uma falha (CHAIPUNYATHAT et al., 2019).

As principais falhas verificadas em projetos de software são relativas a problemas na especificação de requisitos, devido principalmente às dificuldades no entendimento das necessidades do usuário (KOSCIANSKI; SOARES, 2007). Portanto, realizar corretamente

o levantamento, especificação e gerenciamento de requisitos é essencial para garantir o controle de qualidade do software.

Para o controle de qualidade, a verificação e validação de requisitos são cruciais, porque os erros identificados nos requisitos em fases posteriores ao desenvolvimento de software tem um custo maior para serem corrigidas (BILAL et al., 2016). Diversas técnicas foram utilizadas para verificação de requisitos, dentre elas estão as inspeções. Segundo Fogelström e Gorschek (2007), as inspeções provaram ser um meio eficaz de encontrar falhas em diferentes artefatos de software.

Sendo assim, estamos propondo neste estudo uma adaptação da técnica de Leitura Baseada em Perspectiva ou *Perspective-Based Reading* (PBR) (BASILI et al., 1996; SHULL; RUS; BASILI, 2000) especificamente para Sistemas Multiagentes. Esta adaptação da técnica tem como objetivo garantir a qualidade de Documentos de Especificação de Requisitos de Software (DERS). Estes documentos são produzidos com base na notação da *Multi-Agent Systems Requirements Modeling Language* (MASRML) proposta por Guedes et al. (2020). A notação MASRML utiliza alguns conceitos, como por exemplo Casos de Uso Internos ou *Internal Use Cases* (IUC) e Atores de Papéis de Agente ou *AgentRoleActors*. Além disso, esta notação procura suportar o modelo crença–desejo–intenção ou *Belief-Desire-Intention* (BDI) (BRATMAN et al., 1987).

Para executar a inspeção proposta neste estudo estendemos o modelo (*template*) do padrão ISO/IEC/IEEE 29148:2018 (ISO/IEC/IEEE..., 2018). Esta extensão permite a produção de Documentos de Especificação de Requisitos de Software (DERS) que suportem características de sistemas multiagentes.

Um padrão de documentação de requisitos é muito importante pois fornece requisitos de qualidade, concisos, legíveis e fáceis de entender (VIE JR., 2021). Além disso, todas as partes envolvidas se beneficiam dos padrões considerando que há um entendimento comum entre as pessoas que trabalham nos requisitos (FRANCH et al., 2021). Deste modo, garantir que o documento de especificação de requisitos de software tenha a qualidade necessária é fundamental para o sucesso de qualquer projeto de desenvolvimento de software (SAAVEDRA; BALLEJOS; ALE, 2013).

No contexto deste estudo, propomos uma técnica de verificação, considerando a definição apresentada no estudo de Ryan e Wheatcraft (2017), em que se analisou o uso dos termos verificação e validação e suas definições de acordo com a literatura. Neste estudo o autor define que o foco da verificação está no texto e na estrutura do requisito, determinando se ele está redigido ou estruturado corretamente de acordo com o padrões, diretrizes, regras e listas de verificação da organização. Por outro lado, a validação de requisitos permite determinar se os requisitos estão definidos de forma clara e comunicam corretamente as expectativas e necessidades das partes interessadas. Envolve a análise sobre se está sendo construída a coisa certa (conforme definido pelo conjunto de requisitos).

A técnica de verificação proposta neste trabalho é específica para uso em um processo de Engenharia de Requisitos para sistemas multiagentes atualmente em desenvolvimento. No entanto, acreditamos que a técnica proposta possa ser adaptada para uso em outros processos de engenharia de requisitos ou outros artefatos produzidos pela engenharia de requisitos.

1.1 Motivação

Problemas de inconformidades nas especificações de requisitos resultam em software de baixa qualidade, tempo de desenvolvimento mais longo e custos de desenvolvimento elevados (WIDODO et al., 2021). Segundo Iqbal et al. (2020), a principal razão para as falhas em projetos de software parece ser os requisitos inadequados e não validados nos estágios iniciais do desenvolvimento.

Para resolver estes problemas, a verificação e validação de requisitos garante que os requisitos descritos na especificação de requisitos de software sejam completos, consistentes e que estejam de acordo com as necessidades do cliente. Ela garante a validade dos requisitos do usuário, eliminando ambiguidades e inconsistências da especificação de requisitos de software (BILAL et al., 2016). Além disso, segundo Iqbal et al. (2020), a validação de requisitos contribui significativamente para o sucesso dos projetos de software.

No entanto, as técnicas de verificação e validação utilizadas em softwares tradicionais não podem ser totalmente adotadas para o contexto de sistemas multiagentes, tendo em vista que, segundo Fuentes-Fernández, Gómez-Sanz e Pavón (2004) os novos conceitos e propriedades que surgiram com o uso do paradigma de agente, implicam na construção de novas abordagens de verificação e validação, diferentes da engenharia de software convencional.

Sendo assim, a nossa questão de pesquisa geral está relacionada a se é possível desenvolver uma técnica que permita verificar a especificação de requisitos para sistemas multiagentes.

1.2 Objetivos

Este trabalho tem como objetivo geral propor uma técnica de inspeção para verificar documentos de especificação de requisitos para sistemas multiagentes. Para atingir o objetivo geral desta pesquisa, devem ser alcançados os seguintes objetivos específicos:

- estabelecer o estado da arte dos estudos que propõem processos ou extensões de processos de Sistemas Multiagentes que abordem a Engenharia de Requisitos com ênfase na especificação e verificação e validação de requisitos;
- analisar de que forma os estudos apresentam a subárea de validação de requisitos e quais técnicas de verificação são utilizadas;

- analisar de que forma os estudos apresentam a especificação de características do modelo BDI;

- desenvolver uma técnica para inspeção de documentos de especificação de requisitos para sistemas multiagentes;

- desenvolver uma especificação de requisitos que permita produzir documentos com base na notação da *Multi-agent Systems Requirements Modeling Language* (MASRML) proposta por Guedes et al. (2020);

- avaliar a proposta de inspeção quanto à eficiência e eficácia na detecção de defeitos em documentos de especificação de requisitos.

1.3 Contribuições

As principais contribuições deste trabalho são as seguintes:

- uma revisão sistemática apresentando o estado da arte de processos que trabalhem a engenharia de requisitos para sistemas multiagentes com ênfase nas subáreas de especificação e validação;

- uma nova abordagem para inspeção de documentos de especificação de requisitos para sistemas multiagentes por meio da adaptação da técnica de Leitura Baseada em Perspectiva ou *Perspective-Based Reading* (PBR) (BASILI et al., 1996; SHULL; RUS; BASILI, 2000);

- proposta de um Guia de Inspeção de Requisitos para Sistemas Multiagentes (GIRSMA);

- proposição de um processo de inspeção de requisitos para sistemas multiagentes;

- uma especificação de requisitos com base em um padrão reconhecido internacionalmente suportando o modelo BDI e a notação proposta pela MASRML (GUEDES et al., 2020).

A proposta de inspeção apresentada neste estudo constitui uma nova abordagem para verificar requisitos de sistemas multiagentes. Uma vez que, até onde vai nosso conhecimento da literatura, nenhum outro estudo propõe uma técnica de inspeção específica para sistemas multiagentes. Além disso, a especificação de requisitos proposta procura representar as características necessárias ao modelo BDI, diferente de outros estudos que representam parcialmente este modelo.

1.4 Metodologia

Esta pesquisa é classificada como Pesquisa Exploratória uma vez que buscamos adquirir informações que nos permitem entender melhor determinado assunto para uma possível transferência de conhecimento. Segundo Vieira (2010), este tipo de pesquisa levanta dados e problemas que podem vir a servir de apoio para pesquisas futuras mais avançadas.

Dessa forma, este trabalho iniciou com a execução de uma revisão sistemática da literatura que apresentaremos no Capítulo 3. Essa revisão procurou determinar o estado da arte da Engenharia de Requisitos para Sistemas Multiagentes, por meio da recuperação de estudos que propõem processos ou estendem processos para o contexto de Sistemas Multiagentes e que abordem Engenharia de Requisitos com ênfase na especificação e verificação e validação de requisitos.

Quanto aos procedimentos, nossa pesquisa se classifica como Bibliográfica e Pesquisa Experimental, devido a Revisão Sistemática da Literatura e os quase-experimentos que realizamos. Considerando que, segundo Fontelles et al. (2009), em uma pesquisa experimental o investigador seleciona as variáveis que serão estudadas, define a forma de controle sobre elas e observa os efeitos sobre o objeto de estudo, em condições pré-estabelecidas.

Dessa forma, para avaliar a eficácia e eficiência da técnica de inspeção proposta neste estudo, foram executados dois quase-experimentos. Segundo Wohlin (2012), o quase-experimento é uma investigação empírica semelhante a um experimento, onde a atribuiçao de tratamento aos sujeitos não pode ser realizada de forma aleatória. Além disso, devido ao isolamento social imposto pelas medidas de combate ao avanço da COVID-19, não seria possível reunir os sujeitos presencialmente. Dessa forma, o estudo empírico foi classificado como quase-experimento.

1.5 Estrutura do Documento

O restante deste documeto segue a seguinte organização. O Capítulo 2 apresenta conceitos e ideias necessários para o entendimento da solução proposta. No Capítulo 3 apresentamos a revisão sistemática da literatura, onde foram analisados os processos de desenvolvimento de sistemas multiagentes que suportam a engenharia de requisitos. O Capítulo 4 apresenta nossa adaptação da técnica de Leitura Baseada em Perspectiva para inspeção de documentos de especificação de requisitos. Já o Capítulo 5 apresenta nossa proposta de extensão do modelo (*template*) do padrão ISO/IEC/IEEE 29148:2018 para suportar características de sistemas multiagentes com objetivo de aplicar nossa técnica de inspeção. No Capítulo 6 apresentamos os dois quase-experimentos executados para verificar a eficiência e eficácia da técnica de inspeção proposta neste estudo. E por fim, o Capítulo 7 relata a conclusão sobre a execução deste trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta uma visão geral das teorias básicas a fim de fornecer uma melhor compreensão dos conceitos e ideias descritos na dissertação. Ele está dividido da seguinte forma: na seção 2.1 são apresentados conceitos da engenharia de requisitos; na seção 2.2 é apresentado conceitos de inspeções; a seção 2.3 apresenta características de agentes e sistemas multiagentes; na seção 2.4 são apresentados conceitos do Modelo Crenças-Desejos-Intenções; a seção 2.5 aborda conceitos de Casos de Uso Interno; na seção 2.6 é apresentado conceitos de atores de papéis de agente (*AgentRoleActors*); a seção 2.7 apresenta a técnica de leitura baseada em perspectiva; e por fim, a seção 2.8, apresenta a estrutura do modelo (*template*) do padrão ISO/IEC/IEEE 29148:2018.

2.1 Engenharia de Requisitos

De acordo com Berenbach et al. (2009), os objetivos da Engenharia de requisitos (ER) são: (I) identificar requisitos de software; (II) analisar requisitos para classificá-los e derivar requisitos adicionais, bem como resolver conflitos entre eles; (III) para documentar os requisitos; e (IV) para validar os requisitos documentados.

Em *Engineering Body of Knowledge* ou Guia do Conhecimento da Engenharia de Software (SWEBOK) (BOURQUE; FAIRLEY et al., 2014) – um livro de referência na área – afirma-se que o processo de ER cobre quatro subáreas principais: (I) Elicitação de Requisitos; (II) Análise de Requisitos; (III) Especificação de Requisitos; e (IV) Validação de Requisitos.

A elicitação de requisitos investiga como extrair requisitos e quais são suas origens. A análise de requisitos visa detectar e resolver conflitos entre os requisitos, para descobrir os limites do sistema. A especificação de requisitos, por sua vez, produz documentos de requisitos que podem ser sistematicamente revisados, avaliados e aprovados. Finalmente, a validação de requisitos avalia os documentos de requisitos para garantir que os requisitos sejam compreensíveis, consistentes e completos.

2.1.1 Elicitação de Requisitos

A elicitação de requisitos está preocupada com a origem dos requisitos de software e como coletá-los. É o primeiro estágio na construção de uma compreensão do problema que o software se propõe a resolver (BOURQUE; FAIRLEY et al., 2014).

A elicitação de requisitos é um processo para descoberta e coleta das necessidades dos usuários, dos seus desejos, dos seus problemas e de como é possível solucioná-los (NUSEIBEH; EASTERBROOK, 2000). Segundo Rosa et al. (2018), a elicitação de requisitos é o momento que exige uma intensa comunicação entre os “donos de problemas” e os engenheiros de requisitos.

2.1.2 Análise de Requisitos

A Análise de Requisitos visa detectar e resolver conflitos entre requisitos, descobrir os limites do software e como ele deve interagir com seu ambiente organizacional e operacional, bem como elaborar requisitos de sistema para derivar novos requisitos de software (BOURQUE; FAIRLEY et al., 2014).

A análise de requisitos é um dos processos importantes no gerenciamento do ciclo de vida de desenvolvimento de software (CHEBANYUK; PALAHIN; MARKOV, 2020). Nesse sentido, Suhaib (2019) afirma que a análise de requisitos é uma parte importante da Engenharia de Software tanto para o desenvolvimento de aplicativos do sistema quanto para os requisitos do projeto.

2.1.3 Especificação de Requisitos

A especificação de requisitos do sistema, tem como objetivo fornecer uma descrição do que o sistema deve fazer. Deve descrever todas as entradas e saídas e as relações necessárias entre elas (ISO/IEC/IEEE..., 2018). Envolve a produção de um documento que pode ser sistematicamente revisado, avaliado e aprovado (BOURQUE; FAIRLEY et al., 2014).

Permite a obtenção do detalhamento necessário de cada requisito, de forma que este possa explicitar suficientemente suas características, comprovando sua real utilidade para o projeto (ENGHOLM JÚNIOR, 2010). Isso leva à compreensão do sistema e sua estrutura antes da implementação, fornece a base para o resto das fases de desenvolvimento e atua como um ponto de referência para as partes interessadas do sistema (SLHOUB; CARVALHO; BOND, 2017).

Nessa atividade, a equipe de software deve trabalhar diretamente com os interessados, objetivando absorver informações detalhadas sobre o domínio da aplicação, os serviços que o sistema deve oferecer, sua expectativa de desempenho, as restrições de *hardware*, dentre outros (SOMMERVILLE, 2011).

2.1.4 Verificação de Requisitos

O alinhamento fraco da engenharia de requisitos com a verificação pode levar a problemas na entrega dos produtos de software, tais como falta de cumprimento de prazos e qualidade insuficiente (BJARNASON et al., 2014). De acordo com Dzida e Freitag (1998) a verificação está relacionada com a correção dos requisitos. Além disso, segundo Ryan e Wheatcraft (2017), em geral, a verificação se refere à estrutura do requisito sendo verificado, certificando-se de que este atende às diretrizes que orientam sua criação, ou seja, as regras sobre como escrever requisitos, padrões e seguindo as melhores práticas de requisitos bem formados.

A verificação de requisitos é a confirmação por exame de que os requisitos (individualmente ou em conjunto) são bem formados (ISO/IEC/IEEE..., 2011; ISO/IEC/IEEE..., 2018). Segundo Ryan e Wheatcraft (2017), isso significa que um requisito ou conjunto de requisitos foi revisado para garantir que as características de bons requisitos sejam alcançadas.

Segundo Khan et al. (2015), os especialistas acreditam que revisão e inspeção são as melhores estratégias para remover ou minimizar os desafios da verificação e validação de requisitos. Nesse sentido, Ryan e Wheatcraft (2017) afirmam que a verificação de requisitos confirma, por inspeção, que os requisitos contêm os elementos necessários e possuem as características de um requisito bem formado.

2.1.5 Validação de Requisitos

A validação de requisitos se preocupa em demonstrar que os requisitos definem o sistema que o cliente realmente deseja. Durante esta etapa é verificado se o requisito é válido, se o requisito é consistente, se o requisito está completo, se o requisito pode ser implementado pela arquitetura disponível e dentro do prazo e custo estipulados, e se o requisito pode ser verificável (SOMMERVILLE, 2007).

Segundo Pressman (2011), na etapa de validação dos requisitos, qualquer artefato produzido pela engenharia de requisitos de determinado sistema deve ser avaliado quanto à qualidade, tendo sua especificação validada para garantir que todos os requisitos tenham sido declarados de maneira não ambígua e que inconsistências, omissões e erros tenham sido detectadas e corrigidas.

A validação de requisitos é uma fase necessária da engenharia de requisitos e desempenha um papel crucial para que qualquer projeto seja bem sucedido. Ela visa garantir que as necessidades dos clientes sejam completas e bem escritas (GUPTA; DERAMAN; SIDDIQUI, 2019).

Validação de requisitos é a confirmação, por exame, de que os requisitos (individualmente e como um conjunto) definem o sistema conforme pretendido pelas partes interessadas (ISO/IEC/IEEE..., 2011; ISO/IEC/IEEE..., 2018). Nesse sentido, o PMBOK (PMI, 2013) define que a validação de requisitos visa garantir que um produto, serviço ou sistema atende às necessidades do cliente e outras partes interessadas identificadas.

2.2 Inspeções

Várias técnicas de verificação e validação de requisitos foram aplicadas para desenvolver software de qualidade, como revisão de requisitos, inspeções, prototipagem, baseado em modelo, teste de requisitos e validação de requisitos orientada por ponto de vista (SAQI; AHMED, 2008; RAJA, 2009; YOUSUF; ZAMAN; IKRAM, 2009).

Segundo Ebad (2017), umas das técnicas de verificação de requisitos que se destaca

é a inspeção. Esse processo requer uma agenda que oriente a preparação da revisão e a própria reunião. Ela tem requisitos de entrada e saída rigorosos para as entregas de trabalho do projeto.

A inspeção pode ser aplicada a qualquer artefato produzido durante o ciclo de vida de desenvolvimento de software, incluindo as representações mais legíveis do software, como seus requisitos ou um modelo de projeto (LEWIS, 2009). Dessa forma, a inspeção permite detectar requisitos inconsistentes durante a especificação de requisitos (EBAD, 2017).

A inspeção de software é um método comprovado que permite a detecção e remoção de inconformidades em artefatos de software assim que esses artefatos são criados. Esse método, geralmente envolve atividades nas quais uma equipe de pessoal qualificado determina se o artefato criado é de qualidade suficiente. As deficiências de qualidade detectadas são corrigidas posteriormente (LAITENBERGER, 2001). Dessa forma, as inspeções de software são um meio eficiente de melhorar a qualidade (CARNEIRO et al., 2017).

2.3 Agentes e Sistemas Multiagentes

Um agente é um processo situado em um ambiente, projetado para atingir um propósito por meio de um comportamento autônomo e flexível. O ambiente é o domínio da aplicação onde o agente irá atuar (VICARI; GLUZ, 2007).

Um Sistema Multiagentes (SMA) consiste em um conjunto de agentes que podem interagir entre si. Agentes são capazes de atuar em um ambiente e ter diferentes "esferas de influência", podendo controlar ou influenciar diferentes partes do ambiente (WOOLDRIDGE, 2009).

2.4 Modelo Crenças-Desejos-Intenções

As crenças-desejos-intenções ou *Belief-Desire-Intention* (BDI) é um modelo de software desenvolvido para programar agentes inteligentes. Inclui crenças, desejos e intenções na arquitetura do agente (BRATMAN et al., 1987).

As crenças representam o estado de informação que o agente possui, ou seja, o que ele acredita ser verdade sobre o meio ambiente, sobre si mesmo e sobre outros agentes. Os desejos representam o estado motivacional dos agentes. Eles representam os objetivos ou situações que o agente deseja alcançar. Finalmente, as intenções representam os desejos que o agente acredita que pode alcançar e realiza ações para alcançá-los (RAO; GEORGEFF, 1995).

Este modelo permite aos agentes um comportamento mais complexo do que os modelos reativos, sem a sobrecarga computacional das arquiteturas cognitivas. Além disso, é mais fácil especificar o conhecimento por meio deste modelo (LARSEN, 2018).

Ele é um dos modelos mais populares para o desenvolvimento de agentes racionais, baseados em como os humanos agem de acordo com as informações derivadas de um ambiente (UJJWAL; CHODOROWSKI, 2019).

Segundo Herzig et al. (2017), os conceitos de crença e objetivo desempenham um papel central na concepção e implementação de agentes autônomos. O conceito de BDI, considera as atitudes mentais fundamentais para os agentes, onde as crenças são adaptadas às verdades do ambiente, enquanto nas intenções os agentes procuram fazer com que o ambiente corresponda aos seus objetivos.

2.5 Caso de Uso Interno

Os Casos de Uso Internos ou *Internal Use Case* (IUC) são casos de uso que não são acessados por atores externos, eles se diferenciam dos casos de uso normais por representarem funcionalidades internas que não podem ser acessadas pelos usuários (eles sequer têm consciência de sua existência). IUCs são acessados por Papéis de Agente (*AgentRoleActors*). Já os casos de uso normais são acessados por atores externos (fora da fronteira do sistema representados por atores normais). Os casos de uso internos possuem estereótipos do tipo *Goal, Plan, Perception ou Action.*

Segundo Yu (1996), um objetivo ou *Goal* (*Desire*) é descrito como uma condição ou estado do mundo que o ator deseja alcançar. Já segundo Morreale et al. (2006), os objetivos ou *Goals* podem ser usados como uma abstração para modelar as funções em torno das quais os sistemas podem selecionar autonomamente o comportamento adequado.

Segundo DeLoach e Garcia-Ojeda (2014), um plano (*Plan*) é responsável por capturar como um agente pode atingir um tipo específico de objetivo usando um conjunto de ações (que inclui o envio e recebimento de mensagens). Já segundo Russell e Norvig (2016), um plano de agente pode ser definido como uma forma do agente atingir seus objetivos. É composto por ações que podem causar transições nos estados do ambiente.

A percepção (*Perception*) fornece aos agentes informações sobre o mundo em que eles habitam. A percepção é causada por um ou mais sensores. Um sensor é qualquer coisa que possa alterar o estado computacional do agente em resposta a uma mudança no estado do ambiente. Um sensor pode ser tão simples quanto um bit que representa se um interruptor está ligado ou desligado, ou tão complexo quanto a retina do olho humano, a qual contém mais de cem milhões de elementos fotossensíveis (RUSSELL; NORVIG, 2016).

Segundo Choren e Lucena (2005), uma ação (*Action*) é um ato computacional que resulta em uma alteração no estado do ambiente. Podem haver dois tipos de ação, a saber: I) ação direta: ação realizada enquanto o agente participa de um cenário para atingir um objetivo; e II) ação adaptativa: ação em que, no meio de um determinado cenário, é necessário que o agente se adapte e essa adaptação requer alguma forma de raciocínio.

2.6 *AgentRoleActor*

Os Atores de Papéis de Agentes ou *AgentRoleActors* representam papéis assumidos por agentes que podem ter estereótipos do tipo *ReactiveAgentRoles* ou *CognitiveAgentRoles*, esses papéis de agentes são modelados dentro da fronteira do sistema. Os papéis de agente representam as funções que os agentes podem desempenhar dentro de um sistema. Os agentes podem assumir mais de um papel, mas geralmente não ao mesmo tempo.

2.7 Leitura Baseada em Perspectiva

A Leitura Baseada em Perspectiva ou *Perspective-Based Reading* (PBR) (BASILI et al., 1996; SHULL; RUS; BASILI, 2000), fornece um conjunto de procedimentos que podem ajudar os desenvolvedores a resolver problemas de inspeção de requisitos de software. Os revisores da PBR representam partes interessadas específicas no documento (como projetistas ou testadores) para verificar a qualidade dos requisitos especificados (SHULL; RUS; BASILI, 2000). Segundo Ebad (2017), a PBR é considerada uma das técnicas mais eficazes para inspecionar os requisitos.

A PBR é amplamente aplicável e personalizável a situações particulares, não é estritamente formalizada em um conjunto definitivo de procedimentos. Dessa forma, PBR envolve basicamente: I) selecionar um conjunto de perspectivas para revisar o documento de requisitos; II) criar ou adaptar procedimentos para cada perspectiva utilizável para construir um modelo das informações de requisitos relevantes; III) aumentar cada procedimento com perguntas para encontrar defeitos ao criar o modelo; e IV) aplicar os procedimentos de revisão do documento (SHULL; RUS; BASILI, 2000).

A técnica PBR fornece questões desenvolvidas especificamente para cada passo do procedimento e Dessa forma cria representações para que os revisores possam responder uma série de questões sobre o produto do trabalho (PAGLIUSO; TAMBASCIA; VILLAS-BOAS, 2002). Cada inspetor deve definir o seu interesse no documento de forma que esse seja inspecionado de acordo com a perspectiva de uma parte envolvida específica.

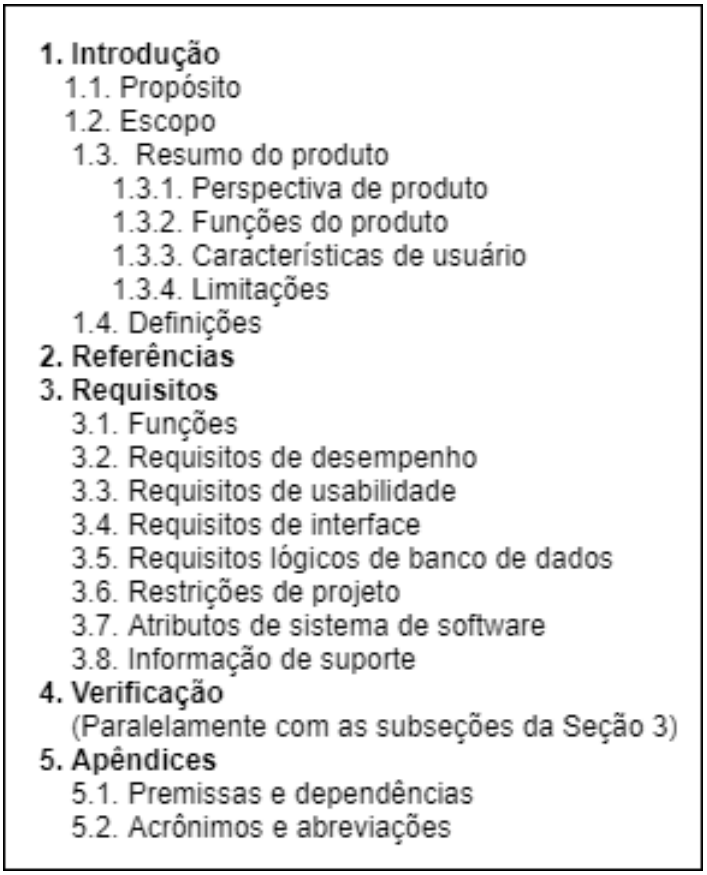
2.8 Padrão ISO/IEC/IEEE 29148:2018

O padrão ISO/IEC/IEEE 29148:2018, fornece diretrizes para os processos e produtos na engenharia de requisitos ao longo do ciclo de vida dos sistemas e softwares. Um dos processos neste ciclo de vida é a Especificação de Requisitos de Software (ERS), que provê uma coleção estruturada dos requisitos essenciais, considerando funções, desempenho, restrições de projeto, atributos do software e suas interfaces externas (ISO/IEC/IEEE..., 2018).

O padrão ISO/IEC/IEEE 29148:2018 sugere um modelo (*template*) que compreende uma estrutura básica para produzir um documento de especificação de requisitos de

software. Esta estrutura está compreendida em 5 seções, dentre as quais nos concentramos na seção de *Requirements* ou Requisitos (seção 3). Nessa seção devem ser descritos as funções do produto de software, os requisitos de desempenho, requisitos de usabilidade, requisitos de lógica de base de dados, restrições de projeto, atributos de sistemas e *softwares* e informações de suporte. A estrutura básica do modelo (*template*) do padrão pode ser observada na Figura 1.

Figura 1 – Estrutura básica do modelo do padrão ISO/IEC/IEEE 29148:2018.



Fonte: ISO/IEC/IEEE... (2018)

.

Conforme pode ser observado na Figura 1, a seção 3 do modelo do padrão ISO/IEC/-IEEE 29148:2018 tradicional não apresenta características específicas para sistemas multiagentes.

No estudo de Vie Jr. (2021), ele sugere que um padrão de especificação de requisitos deve ser utilizado como guia para desenvolvimento de modelos (*templates*) próprios. Esses modelos (*templates*) adaptados visam atender a especificação de requisitos em projetos específicos.

3 TRABALHOS RELACIONADOS

Este Capítulo tem como objetivo apresentar os estudos que serviram como base ou referência para a realização deste trabalho. Para determinar esses estudos foi realizada uma revisão sistemática de literatura com o objetivo de analisar os processos ou extensões de processos de desenvolvimento para SMAs que suportem a etapa de ER com ênfase na Especificação e Verificação e Validação.

Assim sendo, a seção 3.1 apresenta de que forma foi realizada a revisão sistemática, compreendendo deste o protocolo até os resultados obtidos. Já a seção 3.2 apresenta outros estudos que foram localizados posteriormente a execução da revisão sistemática da literatura.

3.1 Revisão Sistemática da Literatura

A Revisão Sistemática da Literatura (RSL) é uma técnica de pesquisa que tem por objetivo identificar, selecionar, avaliar, interpretar e resumir os estudos disponíveis considerados relevantes para o tema de pesquisa ou fenômeno de interesse (KITCHENHAM; CHARTERS, 2007). Essa técnica busca estudos primários relacionados ao tema e fornece uma síntese mais aprofundada sobre os dados obtidos nesses estudos (KITCHENHAM; BRERETON, 2013).

Uma RSL tem como base um protocolo previamente definido, que formaliza sua execução, iniciando pela estipulação das questões de pesquisa, passando pelo estabelecimento dos critérios de inclusão e exclusão dos estudos, selecionando a base digital para a extração dos trabalhos relacionados às palavras-chave aplicadas durante a busca nestas bases, e concluindo com a definição de como os resultados serão apresentados (BIOLCHINI et al., 2005).

Nossa revisão teve como objetivo estabelecer o estado da arte dos processos / metodologias de desenvolvimento de SMA que auxiliam de alguma forma na engenharia de requisitos para este tipo de sistema. Nosso principal interesse é sobre como esses processos identificam e especificam os recursos do modelo BDI na fase de engenharia de requisitos e de que forma validam os artefatos produzidos na especificação. Esta RSL e parte de seus resultados foram publicados pelos autores Mendonça, Souza Filho e Guedes (2021).

3.1.1 Questões de Pesquisa

Definimos seis Questões de Pesquisa (QPs) para esta revisão. A primeira questão de pesquisa (QP1) visa identificar quais metodologias / processos suportam ER para SMA.

A segunda questão de pesquisa (QP2) foi definida para identificar a abrangência da ER por essas metodologias. Acreditamos que com esta questão podemos descobrir

possíveis lacunas na área e que isso permitirá pesquisas futuras.

A terceira questão de pesquisa (QP3) visa verificar quais metodologias suportam o modelo BDI. Este é um modelo consolidado no desenvolvimento do SMA e acreditamos que agrega maior confiabilidade no uso das metodologias que o suportam.

A quarta questão de pesquisa (QP4) tem como objetivo discutir os estudos que cobrem a subárea de especificação de requisitos e de que forma os requisitos são representados.

A quinta questão de pesquisa (QP5) tem como objetivo discutir os estudos que cobrem a subárea de validação de requisitos e quais técnicas são utilizadas.

Por fim, a sexta questão de pesquisa (QP6) tem como objetivo mostrar uma visão mais ampla das necessidades da área e focar nos pontos que podem ser abordados em trabalhos futuros.

As cinco questões de pesquisa estão listadas abaixo:

- QP1: Quais metodologias para o desenvolvimento de SMA suportam um ciclo de vida de engenharia de requisitos ER específico para este tipo de sistema?
- QP2: Qual é a cobertura da engenharia de requisitos por essas metodologias tomando como base as subáreas definidas por SWEBOK (BOURQUE; FAIRLEY et al., 2014)?
- QP3: Qual dessas metodologias enfoca o modelo BDI durante a engenharia de requisitos?
- QP4: Quais dessas metodologias suportam a subárea de especificação de requisitos e de que forma ela está estruturada?
- QP5: Quais dessas metodologias suportam a subárea de validação de requisitos e quais técnicas são utilizadas?
- QP6: Quais são as lacunas existentes nas metodologias que suportam ER para SMA?

3.1.2 Identificação e Seleção de Estudos Primários

Para recuperar trabalhos relevantes para este estudo, construímos um *String* contendo um conjunto de palavras-chave com base nas questões de pesquisa. Esta *String* foi adaptada às particularidades de cada base bibliográfica.

Para realizar esta revisão, utilizamos bases bibliográficas que (I) possuem mecanismo de busca baseado na *web*; (II) possuem mecanismo capaz de utilizar palavras-chave; (III) conter documentos da área de ciência da computação; e (IV) suas bases de dados são atualizadas regularmente. É importante destacar que não limitamos o período em que os estudos foram publicados.

Além disso, incluímos um livro (COSSENTINO et al., 2014c) sobre metodologias para SMA, bem como outros estudos clássicos e conhecidos. Esses estudos foram selecionados manualmente por um especialista da área, pois consideramos que não seriam selecionados na *String* de busca por não apresentarem em seu título, resumo e palavras-chave tópicos relacionados à engenharia de requisitos e por não serem processos focados em ER, embora seus ciclos de vida englobem a área de ER.

Na Tabela 1 mostramos as bases bibliográficas utilizadas, as *Strings* adaptadas para cada base e os filtros selecionados.

Além da *String* de busca, utilizamos filtros manuais nas bases bibliográficas. Os filtros podem ser observados na Tabela 1. Consideramos necessário aplicar esses filtros manuais porque, em algumas bases, os resultados obtidos foram elevados e muitos dos estudos devolvidos estavam fora do escopo.

3.1.3 Critérios de Inclusão e Exclusão

Os critérios de seleção têm como objetivo identificar os estudos primários que fornecem conteúdos para responder às questões de pesquisa. Assim, inicialmente os estudos foram analisados com base no título, resumo e palavras-chave. Caso ainda houvesse dúvidas quanto à classificação final de um estudo em relação aos critérios de inclusão ou exclusão, um especialista seria consultado. Esses critérios estão descritos na Tabela 2.

3.1.4 Avaliação de Qualidade dos Estudos

Definimos dois Critérios de Qualidade (CQs) para avaliar a relevância dos estudos para o escopo desta pesquisa. Esses critérios não foram utilizados para a exclusão de estudos, apenas para a classificação dos estudos mais relevantes. A seguir descrevemos os dois critérios qualitativos e a pontuação atribuída para cada um dos critérios definidos.

1. CQ1: O trabalho suporta o modelo BDI?
   - **Sim (S)**: o trabalho suporta totalmente o modelo BDI;
   - **Parcialmente (P)**: o trabalho suporta pelo menos uma das características do modelo BDI;
   - **Não (N)**: o estudo não suporta o modelo BDI.
2. CQ2: O trabalho aplica algum estudo empírico (experimento, estudo de caso, etc.)?
   - **Sim (S)**: o trabalho aplica algum estudo empírico;
   - **Não (N)**: o trabalho não aplica algum estudo empírico.

Para estabelecer um índice geral de qualidade dos estudos selecionados, atribuímos pontuação a cada critério definido, sendo que Sim (S) corresponde a 1 ponto, Parcialmente (P) 0.5 pontos e para Não (N) é atribuído 0.

Tabela 1 – *Strings* e filtros aplicados nas bases bibliográficas.

Base	**String**	**Filtro**
ACM Library	(“multiagent” OR “multi-agent” OR “multi agent” OR “agent-based” OR “agent society”) AND (“Methodology” OR “method” OR “process”) AND (“requirements engineering” OR “requirements elicitation” OR “requirements modeling” OR “requirements analysis” OR “requirements specification”)	Title/Abstract/keywords
Engineering Village	(“multiagent” OR “multi-agent” OR “multi agent” OR “agent-based” OR “agent society”) AND (“Methodology” OR “method” OR “process”) AND (“requirements engineering” OR “requirements elicitation” OR “requirements modeling” OR ‘requirements analysis” OR “requirements specification”)	Subject/Title/Abstract
IEEE Xplore	(“multiagent” OR “multi-agent” OR “multi agent” OR “agent-based” OR “agent society”) AND (“Methodology” OR “method” OR “process”) AND (“requirements engineering” OR “requirements elicitation” OR “requirements modeling” OR “requirements analysis” OR “requirements specification”)	All metadata, filters suggested by the base software agents and multi-agent systems
Science Direct	(“multiagent” OR “multi-agent” OR “agent-based”) AND (“Methodology” OR “method” OR “process”) AND (“requirements engineering” OR “requirements elicitation” OR “requirements modeling” OR “requirements analysis” OR “requirements specification”)	Title/Abstract/keywords and commands “multiagent” OR multi-agent OR “agent-based”
Scopus	(“multiagent” OR “multi-agent” OR “multi agent” OR “agent-based” OR “agent society”) AND (“Methodology” OR “method” OR “process”) AND (“requirements engineering” OR “requirements elicitation” OR “requirements modeling” OR “requirements analysis” OR “requirements specification”)	Title/Abstract/keywords
Springer Link	((“multiagent” OR “multi-agent” OR “multi agent” OR “agent-based” OR “agent society”) AND (“Methodology” OR “method” OR “process”) AND (“requirements engineering” OR “requirements elicitation” OR “requirements modeling” OR “requirements analysis” OR “requirements specification”))	Filter of the area: Computer science; Filter of the subarea: Software Engineering and Artificial intelligence

Fonte: Autor.

A Tabela 3, mostra a pontuação de cada estudo selecionado. Notamos que apenas três estudos ((JO; EINHORN, 2005), (MYLOPOULOS; CASTRO; KOLP, 2013) e (MORREALE et al., 2006)) alcançaram a classificação máxima de 2 pontos.

Por outro lado, alguns estudos obtiveram pontuação 0 ((ALONSO et al., 2004), (HAJER; TAIEB; RAOUF, 2009), (BOKMA et al., 1994), e (GONZÁLEZ-MORENO et al., 2014)), embora esses estudos não atingir qualquer pontuação, foram mantidos porque os critérios qualitativos foram utilizados apenas para ranquear os estudos, não para eliminá-los.

Tabela 2 – Critérios de Inclusão e Exclusão.

Critério	**ID**	**Descrição**
Inclusão	CI1	O estudo apresenta uma metodologia ou uma extensão de uma metodologia para sistemas multiagentes que contemple pelo menos uma das subáreas de engenharia de requisitos definidas no SWEBOK?
Exclusão	CE1	Estudos que abrangem uma metodologia já incluída em trabalhos mais recentes.
	CE2	Estudos que não são um artigo ou capítulo de livro.
	CE3	Estudos com menos de 6 páginas.
	CE4	Estudos que se resumem a um estudo de caso ou avaliação de metodologia.
	CE5	Estudos que se resumem a uma comparação de metodologias.
	CE6	Estudos que não apresentam metodologia (ou extensão de metodologia) para sistemas multiagentes que contemplem pelo menos uma das subáreas da engenharia de requisitos do SWEBOK.
	CE7	Estudos que se concentram em outras áreas da Engenharia de Software.
	CE8	Estudos que se resumem ao desenvolvimento de um sistema.
	CE9	Estudos que apresentam uma metodologia ou extensão de uma metodologia criada apenas para um tipo de aplicação específica.

Fonte: Autor.

3.1.5 Estratégia de Extração de Dados

Concluído o processo de seleção dos estudos, foram registradas as informações básicas de cada artigo para a extração dos dados. A extração foi realizada por meio de uma planilha do Google para captar todas as informações de cada trabalho incluído, permitindo a síntese posterior. Os dados extraídos dos trabalhos incluídos foram analisados a fim de responder às questões de pesquisa. Na subseção 3.1.8, esses resultados são expostos e discutidos.

3.1.6 Condução da Revisão

A condução desta revisão sistemática foi realizada entre os meses de fevereiro e maio de 2020. Definimos quatro etapas para a seleção dos estudos: (I) execução *String* de busca nas bases bibliográficas; (II) remover os estudos duplicados; (III) aplicação dos critérios de inclusão e exclusão aos trabalhos; e (IV) leitura e extração das informações dos demais estudos da Etapa (III). Os estudos foram lidos por dois revisores em consulta com um especialista na área.

Na Etapa 1, foi realizada a busca de *String* nas bases bibliográficas selecionadas para esta revisão. A visão geral desta etapa pode ser observada na Figura 2. A condução começou analisando os 1060 trabalhos importados das bases bibliográficas selecionadas.

No Estágio 2, um total de 247 estudos duplicados foram removidos. Na Etapa 3, foram aplicados os critérios de inclusão e exclusão a partir da leitura do título, resumo e palavras-chave, resultando na seleção de 53 estudos considerados promissores.

Para evitar a seleção de trabalhos que não se enquadram no escopo desta revisão, os 53 estudos, selecionados na terceira etapa, foram lidos na íntegra, o que resultou na exclusão de 10 trabalhos, totalizando 43 trabalhos selecionados. Os trabalhos rejeitados nessa etapa obedeceram a dois critérios de exclusão:

Tabela 3 – Índices de qualidade dos estudos.

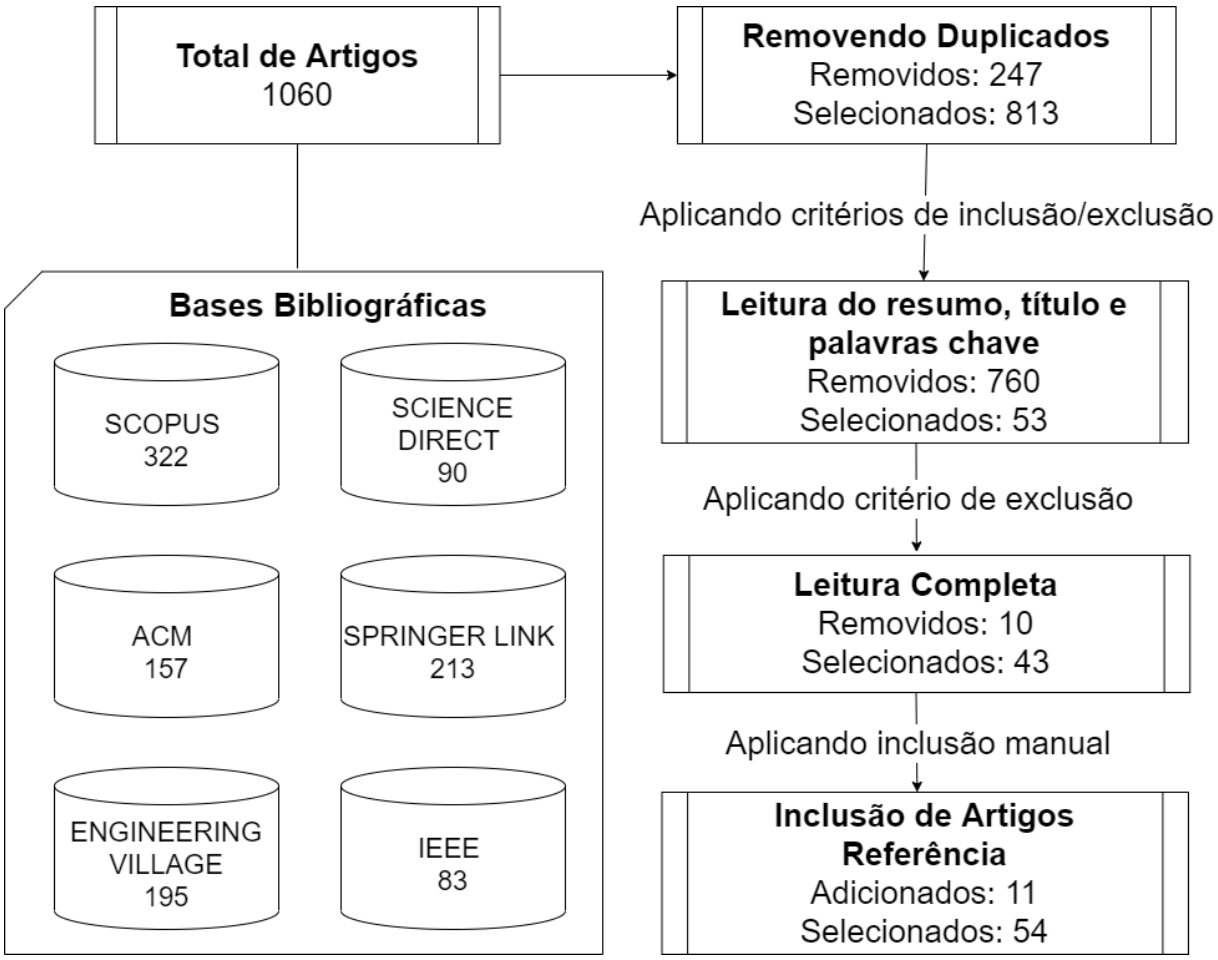
Estudo	CQ1	CQ2	Total
(ULFAT-BUNYADI; MOHAMMADI; HEISEL, 2018)	0.50	0.00	0.50
(ABUSHARK et al., 2016)	0.50	1.00	1.50
(RIBINO et al., 2013)	1.00	0.00	1.00
(LIU et al., 2011)	0.00	1.00	1.00
(ARGENTE; BOTTI; JULIÁN, 2009)	0.50	0.00	0.50
(SEN; HEMACHANDRAN, 2010)	0.50	1.00	1.50
(BLANES; INSFRAN; ABRAHãO, 2009)	0.50	1.00	1.50
(HUIYING; ZHI, 2009)	0.50	0.00	0.50
(FUENTES-FERNÁNDEZ; GÓMEZ-SANZ; PAVÓN, 2009)	0.50	1.00	1.50
(RODRIGUEZ et al., 2009)	0.50	1.00	1.50
(BRYL et al., 2008)	0.50	0.00	0.50
(LEE; LEE, 2008)	0.50	1.00	1.50
(RANJAN; MISRA, 2006a)	0.50	0.00	0.50
(LEE; LEE, 2006)	0.50	0.00	0.50
(SHEN et al., 2005)	0.50	0.00	0.50
(ALONSO et al., 2004)	0.00	0.00	0.00
(JO; EINHORN, 2005)	1.00	1.00	2.00
(MYLOPOULOS; CASTRO; KOLP, 2013)	1.00	1.00	2.00
(CYSNEIROS; YU, 2002)	1.00	0.00	1.00
(LEE; LIU, 2002)	0.50	0.00	0.50
(BANACH, 2010)	0.50	0.00	0.50
(MORREALE et al., 2006)	1.00	1.00	2.00
(MURRAY, 2003)	0.00	1.00	1.00
(COSSENTINO et al., 2010)	0.50	0.00	0.50
(BRESCIANI; DONZELLI, 2003)	0.50	1.00	1.50
(JAIN, 2007)	0.50	1.00	1.50
(HSIEH et al., 2008)	0.00	1.00	1.00
(SUTCLIFFE, 2001)	0.50	1.00	0.00
(SEN; JAIN, 2007)	0.50	1.00	1.50
(HAUMER et al., 1999)	0.50	1.00	1.50
(CAO et al., 2004)	0.50	1.00	1.50
(WILMANN; STERLING, 2005)	0.50	0.00	0.50
(WU; LIU; MA, 2010)	0.50	1.00	1.50
(LIU; LI, 2015)	0.00	1.00	1.00
(HAJER; TAIEB; RAOUF, 2009)	0.00	0.00	0.00
(ASHAMALLA; BEYDOUN; LOW, 2017)	0.50	1.00	1.50
(BOKMA et al., 1994)	0.00	0.00	0.00
(HILAIRE et al., 2013)	0.50	0.00	0.50
(GAUR; SONI, 2012)	0.50	1.00	1.50
(PASSOS; ROSSETTI; GABRIEL, 2015)	0.50	1.00	1.50
(WANG et al., 2013)	0.50	1.00	1.50
(RONALD et al., 2012)	0.50	1.00	1.50
(DOMANN et al., 2014)	0.00	1.00	1.00
(COSSENTINO et al., 2014b)	0.50	0.00	0.50
(COSSENTINO; SEIDITA, 2014)	0.00	1.00	1.00
(GONZÁLEZ-MORENO et al., 2014)	0.00	0.00	0.00
(BONJEAN et al., 2014)	0.00	1.00	1.00
(DELOACH; GARCIA-OJEDA, 2014)	0.50	0.00	0.50
(PADGHAM; WINIKOFF, 2002)	1.00	0.00	1.00
(CAIRE et al., 2001)	0.50	1.00	1.50
(CAO, 2015)	0.50	0.00	0.50
(GLASER, 1997)	1.00	0.00	1.00
(IGLESIAS et al., 1998)	1.00	0.00	1.00
(LIND, 2001)	0.00	1.00	1.00

Fonte: Autor.

1. Trabalhos que se concentrem em outras áreas da Engenharia de Software: os trabalhos excluídos que estavam dentro desse critério foram metodologias que trabalharam com SMA apenas em etapas posteriores à engenharia de requisitos. A engenharia de requisitos foi realizada de forma tradicional, não focando em nenhuma característica particular do SMA.

2. Trabalhos que abrangem uma metodologia já incluída em um trabalho: para esse

Figura 2 – Processo de Pesquisa.



Fonte: Autor.

critério selecionamos os trabalhos mais recentes de forma que possamos entender o estado atual da metodologia.

Ao final da Etapa de condução, os estudos selecionados manualmente ((COSSENTINO et al., 2014b), (COSSENTINO; SEIDITA, 2014), (GONZÁLEZ-MORENO et al., 2014), (BONJEAN et al., 2014), (DELOACH; GARCIA-OJEDA, 2014), (PADGHAM; WINIKOFF, 2002), (CAIRE et al., 2001), (CAO, 2015), (GLASER, 1997), (IGLESIAS et al., 1998), (LIND, 2001)) foram adicionados ao conjunto de artigos pesquisados nas bases, conforme definido na subseção 3.1.2. Isso resultou em um total de 54 estudos aceitos.

3.1.7 Extração dos Dados

Para extração de dados nos trabalhos aceitos, lemos todos e procuramos identificar quais subáreas da ER do SWEBOK cada trabalho cobre, se a metodologia proposta no

estudo tem um ciclo de vida bem definido, se o RE apresentado no estudo é adequado para SMA e se o estudo é compatível com o modelo BDI.

A condução dessa etapa foi realizada em dupla, onde cada pesquisador leu o artigo e extraiu as informações sobre as questões citadas anteriormente. Os conflitos entre os pesquisadores foram resolvidos por um especialista da área.

3.1.8 Resultados

As informações relevantes dos estudos selecionados foram obtidas por meio da planilha de extração de dados. As evidências encontradas sobre cada questão de pesquisa são discutidas nas próximas subseções.

3.1.8.1 Análise das Questões de Pesquisa

Nesta seção, respondemos às questões de pesquisa deste estudo e discutimos os resultados alcançados.

Questão de Pesquisa 1: Quais metodologias para o desenvolvimento de SMA suportam um ciclo de vida de engenharia de requisitos ER específico para este tipo de sistema?

Para responder a esta questão, encontramos 54 metodologias que abordaram ER para SMA. Esses estudos podem ser observados na Tabela 4.

Questão de Pesquisa 2: Qual é a cobertura da engenharia de requisitos por essas metodologias tomando como base as subáreas definidas por SWEBOK (BOURQUE; FAIRLEY et al., 2014)?

Dos 54 estudos selecionados observamos que todos eles apresentam adequação de Engenharia de Requisitos para sistemas multiagentes. Assim, extraímos quais subáreas de ER definidas em SWEBOK (BOURQUE; FAIRLEY et al., 2014) são suportadas por esses estudos.

A Tabela 4 mostra os 54 estudos e as subáreas que os apoiam. Grande parte desses estudos, 45 no total, atende à subárea de análise de requisitos. Enquanto 32 deles suportam a subárea de especificação de requisitos.

A subárea de elicitação de requisitos, por sua vez, está amparada em 17 estudos. Por fim, a subárea de validação de requisitos apresenta o menor número de estudos, com apenas 3 do total.

Observamos também que, a partir desses estudos, apenas a metodologia ADELFE (BONJEAN et al., 2014) suporta as quatro subáreas de ER (elicitação, análise, especificação e validação). Além disso, a elicitação na metodologia ADELFE não é adequada para SMA, sendo aplicada uma elicitação tradicional. Os recursos adequados para um SMA começaram a ser apresentados na fase de análise. No entanto, essa etapa não apresenta os

meios de verificação e validação dos documentos específicos do SMA. ADELFE verifica e valida apenas documentos presentes em uma engenharia de requisitos tradicional.

Outro fato importante que percebemos na extração é que apenas 30 estudos apresentaram alguns experimentos empíricos para a validação da metodologia.

Tabela 4 – Metodologias/Processos que suportam engenharia de requisitos para sistema multiagente e sua cobertura em relação às subáreas do SWEBOK.

Metodologia	**Elicitação**	**Análise**	**Especificação**	**Validação**
KAOS Extension (ULFAT-BUNYADI; MOHAMMADI; HEISEL, 2018)		✓	✓	
JAAMAS (ABUSHARK et al., 2016)			✓	
Patrizia Ribino (RIBINO et al., 2013)			✓	
AGSIRA (LIU et al., 2011)		✓	✓	
GORMAS (ARGENTE; BOTTI; JULIÁN, 2009)		✓	✓	
ATABGE (SEN; HEMACHANDRAN, 2010)	✓			
RE4Gaia (BLANES; INSFRAN; ABRAHãO, 2009)		✓		
Xu Huiying (HUIYING; ZHI, 2009)		✓	✓	
REG for AOSE (FUENTES-FERNÁNDEZ; GÓMEZ-SANZ; PAVÓN, 2009)	✓	✓		
Extension GAIA (RODRIGUEZ et al., 2009)		✓		
B-Tropos (BRYL et al., 2008)	✓	✓	✓	
JONGWON LEE (LEE; LEE, 2008)		✓		
Prabhat Ranjan (RANJAN; MISRA, 2006a)	✓	✓	✓	
SRAMO (LEE; LEE, 2006)		✓		
Zhiqi Shen (SHEN et al., 2005)		✓		
SONIA (ALONSO et al., 2004)		✓	✓	
BDI ASP (JO; EINHORN, 2005)		✓		
Tropos (MYLOPOULOS; CASTRO; KOLP, 2013)	✓	✓	✓	
Cysneiros (CYSNEIROS; YU, 2002)		✓	✓	✓
Chiung-Hui (LEE; LIU, 2002)	✓	✓		
KAOS (BANACH, 2010)		✓	✓	
PRACTIONIST (MORREALE et al., 2006)		✓	✓	
Murray (MURRAY, 2003)			✓	
ASPECS (COSSENTINO et al., 2010)		✓	✓	
REF (BRESCIANI; DONZELLI, 2003)		✓	✓	
Sen and Jain (JAIN, 2007)	✓			
Hsieh et al. (HSIEH et al., 2008)		✓	✓	
Sutcliffe (SUTCLIFFE, 2001)		✓	✓	
Agile Sen and Jain (SEN; JAIN, 2007)	✓			
CREWS-EVE (HAUMER et al., 1999)	✓		✓	✓
Cao et al. (CAO et al., 2004)		✓	✓	
HOMER (WILMANN; STERLING, 2005)	✓			
Wu et al. (WU; LIU; MA, 2010)	✓			
Liu and Li (LIU; LI, 2015)		✓		
Mahmoud et al. (HAJER; TAIEB; RAOUF, 2009)		✓		
Ashamalla et al. (ASHAMALLA; BEYDOUN; LOW, 2017)	✓	✓		
Consensus (BOKMA et al., 1994)		✓	✓	
Hilaire et al. (HILAIRE et al., 2013)		✓		
Gaur and Soni (GAUR; SONI, 2012)		✓		
Passos et al. (PASSOS; ROSSETTI; GABRIEL, 2015)	✓	✓		
PLANT (WANG et al., 2013)		✓	✓	
Ronald et al. (RONALD et al., 2012)		✓	✓	
aMIAC (DOMANN et al., 2014)		✓	✓	
GAIA (COSSENTINO et al., 2014b)		✓	✓	
PASSI (COSSENTINO; SEIDITA, 2014)	✓	✓	✓	
INGENIAS-SCRUM (GONZÁLEZ-MORENO et al., 2014)	✓	✓	✓	
ADELFE (BONJEAN et al., 2014)	✓	✓	✓	✓
O-MaSE (DELOACH; GARCIA-OJEDA, 2014)	✓	✓	✓	
PROMETHEUS (PADGHAM; WINIKOFF, 2002)		✓		
MESSAGE (CAIRE et al., 2001)		✓	✓	
OSOAD (CAO, 2015)		✓		
COMOMAS (GLASER, 1997)		✓		
MAS-COMMONKADS (IGLESIAS et al., 1998)		✓	✓	
MASSIVE (LIND, 2001)	✓	✓	✓	
Total	17	45	32	3

Fonte: Autor.

Questão de Pesquisa 3: Qual dessas metodologias enfoca o modelo BDI durante a engenharia de requisitos?

Tentamos identificar quais metodologias suportam o modelo BDI. Observamos que a maior parte dos estudos, 35 no total, apoiam parcialmente o modelo BDI, ou seja, eles identificam pelo menos uma das características deste modelo.

Essas características são: crenças do agente; objetivos (desejos) do agente; e intenções do agente. No entanto, é necessário afirmar que a maioria desses trabalhos não cita explicitamente o modelo BDI, a maioria deles são metodologias orientadas a objetivos, ou seja, eles se concentram em apenas uma característica do modelo BDI e não necessariamente usam esse modelo, mas o fato de que esses estudos identificam uma das características é útil para nossa pesquisa.

Os objetivos dos agentes foram o recurso mais identificado, na maioria dos casos de forma isolada. Existem estudos que identificam intenções, porém percebemos que as intenções bem como as crenças não são identificadas isoladamente, sempre acompanhadas da identificação de seus objetivos.

Outra questão a ser destacada é que 11 estudos não apresentam suporte ao modelo BDI e apenas 8 apresentam suporte a todos esses recursos em pelo menos uma etapa de sua engenharia de requisitos. A Tabela 5 apresenta a cobertura das metodologias quanto o seu suporte ao modelo BDI.

Questão de Pesquisa 4: Quais dessas metodologias suportam a subárea de especificação de requisitos e de que forma ela está estruturada?

De todos os estudos remanescentes ao final de nossa RSL, observamos que 32 desses estudos suportam a subárea de especificação de requisitos, conforme apresentado na Tabela 4. No entanto, desses estudos apenas 4 suportam as crenças, desejos e intenções. Já os estudos que suportam parcialmente o modelo BDI somam 18 no total.

Desse modo, descreveremos a seguir de que forma a especificação é realizada e os modelos criados nesta subárea com ênfase no modelo BDI, apenas dos estudos que suportam totalmente ou parcialmente este modelo.

Como estamos interessados apenas nas subáreas de especificação e validação de requisitos, descreveremos apenas estas duas subáreas nos estudos retornados na revisão sistemática. Dessa forma, não discutiremos as fases de projeto e desenvolvimento presentes em alguns estudos.

Tabela 5 – Cobertura de metodologias/processos relativos ao suporte do modelo BDI.

Metodologia	Crença	Desejo (Objetivo)	Intenção	Não Suporta
KAOS Extension (ULFAT-BUNYADI; MOHAMMADI; HEISEL, 2018)		✓	✓	
JAAMAS (ABUSHARK et al., 2016)		✓		
Patrizia Ribino (RIBINO et al., 2013)	✓	✓	✓	
AGSIRA (LIU et al., 2011)		✓		
GORMAS (ARGENTE; BOTTI; JULIÁN, 2009)		✓		
ATABGE (SEN; HEMACHANDRAN, 2010)		✓		
RE4Gaia (BLANES; INSFRAN; ABRAHãO, 2009)		✓		
Xu Huiying (HUIYING; ZHI, 2009)		✓	✓	
REG for AOSE (FUENTES-FERNÁNDEZ; GÓMEZ-SANZ; PAVÓN, 2009)		✓		
Extension GAIA (RODRIGUEZ et al., 2009)		✓		
B-Tropos (BRYL et al., 2008)		✓		
JONGWON LEE (LEE; LEE, 2008)		✓		
Prabhat Ranjan (RANJAN; MISRA, 2006a)		✓		
SRAMO (LEE; LEE, 2006)		✓		
Zhiqi Shen (SHEN et al., 2005)		✓		
SONIA (ALONSO et al., 2004)				✓
BDI ASP (JO; EINHORN, 2005)	✓	✓	✓	
Tropos (MYLOPOULOS; CASTRO; KOLP, 2013)	✓	✓	✓	
Cysneiros (CYSNEIROS; YU, 2002)	✓	✓	✓	
Chiung-Hui (LEE; LIU, 2002)		✓		
KAOS (BANACH, 2010)		✓		
PRACTIONIST (MORREALE et al., 2006)	✓	✓	✓	
Murray (MURRAY, 2003)				✓
ASPECS (COSSENTINO et al., 2010)		✓		
REF (BRESCIANI; DONZELLI, 2003)		✓		
Sen and Jain (JAIN, 2007)		✓		
Hsieh et al. (HSIEH et al., 2008)				✓
Sutcliffe (SUTCLIFFE, 2001)		✓		
Agile Sen and Jain (SEN; JAIN, 2007)		✓		
CREWS-EVE (HAUMER et al., 1999)		✓		
Cao et al. (CAO et al., 2004)		✓		
HOMER (WILMANN; STERLING, 2005)		✓		
Wu et al. (WU; LIU; MA, 2010)		✓		
Liu and Li (LIU; LI, 2015)				✓
Mahmoud et al. (HAJER; TAIEB; RAOUF, 2009)				✓
Ashamalla et al. (ASHAMALLA; BEYDOUN; LOW, 2017)		✓		
Consensus (BOKMA et al., 1994)				✓
Hilaire et al. (HILAIRE et al., 2013)		✓		
Gaur and Soni (GAUR; SONI, 2012)		✓		
Passos et al. (PASSOS; ROSSETTI; GABRIEL, 2015)		✓		
PLANT (WANG et al., 2013)		✓		
Ronald et al. (RONALD et al., 2012)		✓		
aMIAC (DOMANN et al., 2014)				✓
GAIA (COSSENTINO et al., 2014b)		✓		
PASSI (COSSENTINO; SEIDITA, 2014)				✓
INGENIAS-SCRUM (GONZÁLEZ-MORENO et al., 2014)				✓
ADELFE (BONJEAN et al., 2014)				✓
O-MaSE (DELOACH; GARCIA-OJEDA, 2014)		✓	✓	
PROMETHEUS (PADGHAM; WINIKOFF, 2002)	✓	✓	✓	
MESSAGE (CAIRE et al., 2001)		✓		
OSOAD (CAO, 2015)		✓		
COMOMAS (GLASER, 1997)	✓	✓	✓	
MAS-COMMONKADS (IGLESIAS et al., 1998)	✓	✓	✓	
MASSIVE (LIND, 2001)				✓
Total	8	43	11	11

Fonte: Autor.

Estudos que suportam parcialmente o modelo BDI:

Apresentamos na Tabela 6 quais artefatos, modelos ou documentos são utilizados para especificação de requisitos e as lacunas de cada estudo que suporta parcialmente o

modelo BDI.

Tabela 6 – Especificação de requisitos e lacunas de cada estudo que suporta parcialmente o modelo BDI.

Metodologia	**Como Especificam Requisitos**	**Lacunas**
KAOS Extension	• Modelo de Objetivos. • Diagrama de Problemas.	• Não cobre as subáreas de Elicitação e Validação. • Não permite identificar as crenças nem as intenções. • Não permite especificar o padrão de comunicação. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos.
JAAMAS	• Diagramas de Atividades. • Cenários. • Árvores de Objetivos.	• Não permite a identificação de crenças. • Não cobre as subáreas de Elicitação, Análise e Validação. • Não identifica os planos necessários para atingir os objetivos. • Não permite identificar os papéis de agentes. • O diagrama de atividades é utilizado apenas com recursos disponibilizados pela própria UML.
Patrizia Ribino	• Modelo de Objetivo. • Modelo de Ontologia.	• Não cobre a subárea de Validação. • Não permite especificar o padrão de comunicação. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos.
Xu Huiying	• Definição Formal de Ator. • Definição Formal de Relação de Dependência. • Definição Formal da Atividade do Ator.	• Não cobre as subáreas de Elicitação e Validação. • Não permite identificar as crenças. • Não permite especificar o padrão de comunicação. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos.
B-Tropos	• Modelo de Ator de Requisitos Iniciais. • Modelo de Objetivo de Requisitos Iniciais. • Diagrama de Ator de Requisitos Tardios. • Diagrama de Objetivo de Requisitos Tardios. • Tabela de Capacidade de Requisitos Tardios.	• Não apresenta nenhuma técnica para validação dos requisitos. • Não permite especificar o padrão de comunicação. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos. • Não fornece recursos para identificação e modelagem de percepções associadas aos agentes.
Prabhat Ranjan	• Modelo de Objetivo Centrado no Sistema. • Modelo de Papéis. • Modelo de Interação. • Modelo de Protocolo.	• Não permite especificar crenças e intenções. • Não cobre a subárea de Validação. • Não permite especificar o padrão de comunicação. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos.
REF	• Modelo de Objetivos. • Modelo Organizacional.	• Não cobre as subáreas de Elicitação e Validação. • Não permite especificar as crenças. • Não permite especificar o padrão de comunicação. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos.
Sutcliffe	• Descrição de Tarefas e Comunicação. • Modelagem de Casos de Uso. • Descrição de Cenários.	• Não cobre as subáreas de Elicitação e Validação. • Não permite especificar as crenças. • Não adapta a UML para modelagem de casos de uso. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos.
MESSAGE	• Modelo Organizacional. • Modelo de Objetivos e Tarefas. • Modelo de Agentes e Papéis. • Modelo de Interação. • Modelo de Domínio.	• Não cobre as subáreas de Elicitação e Validação. • Não permite especificar as crenças. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos. • Não permite especificar o padrão de comunicação.

(continua)

(continuação)

Metodologia	**Como Especificam Requisitos**	**Lacunas**
O-MASE	• Documento de Especificação de Requisitos. • Modelo de objetivos. • Modelo de Domínio. • Modelo organizacional. • Modelo de papéis. • Documento de descrição de papéis. • Modelo de objetivos de papéis.	• Não cobre a subárea de Validação. • Não permite especificar as crenças. • Não permite especificar o padrão de comunicação. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos.
GAIA	• Modelo de Papel. • Modelo de Interação Preliminar. • Modelo Ambiental.	• Não cobre as subáreas de Elicitação e Validação. • Não permite especificar as crenças. • Não permite especificar o padrão de comunicação. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos.
Ronald et al.	• Descrição de conhecimento individual. • Descrição de conhecimento externo. • Descrição de conhecimento coletivo interno. • Descrição de conhecimento coletivo externo.	• Não cobre as subáreas de Elicitação e Validação. • Não permite especificar as crenças. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos. • Não permite especificar o padrão de comunicação.
Cao et al.	• Modelo de dependência organizacional. • Modelo de lógica organizacional.	• Não cobre as subáreas de Elicitação e Validação. • Não permite especificar as crenças e as intenções. • Não permite especificar o padrão de comunicação. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos.
CREWS-EVE	• Cenas em Multimídia. • Modelo Formal Orientado a Agente. • Modelo de objetivo.	• Não cobre a subárea de Análise. • Não apresenta nenhuma técnica para verificar ou validar os requisitos. • Não permite especificar o padrão de comunicação. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos.
GORMAS	• Modelo de Dimensão Funcional. • Documento de Missão Organizacional. • Documento de Partes Envolvidas. • Documento de Condições Ambientais. • Modelo de Dimensão Estrutural. • Modelo de Dimensões Ambientais. • Identificação de Produtos/Serviços. • Descrição de Produtos/Serviços.	• Não cobre as subáreas de Elicitação e Validação. • Não permite especificar as crenças e intenções. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos. • Não permite especificar o padrão de comunicação.
PLANT	• Modelo Orientado ao Processo. • Modelo de Transformação de Objeto. • Modelo Orientado a Agentes. • Modelo Orientado a Objetivo.	• Não cobre as subáreas de Elicitação e Validação. • Não permite especificar as crenças e intenções.
ASPECS	• Descrição de Requisitos do Domínio. • Descrição de Ontologia do Problema. • Identificação de Papéis e Interações. • Descrição de Cenários. • Identificação de Capacidades.	• Não permite especificar crenças e intenções. • Não cobre as subáreas de Elicitação e Validação. • Não permite especificar o padrão de comunicação. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos.
KAOS	• Modelo de Metas. • Modelo de Objetos • Modelo de Responsabilidade.	• Não cobre as subáreas de Elicitação e Validação. • Não permite especificar o padrão de comunicação. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos.

Fonte: Autor.

Conforme pode ser observado nas Tabela 6, no total 18 estudos que cobrem a subárea de especificação de requisitos suportam parcialmente o modelo BDI. A seguir discutiremos em mais detalhes cada um desses estudos.

Os autores Ulfat-Bunyadi, Mohammadi e Heisel (2018), propõem a combinação do uso de modelo de objetivos e diagramas de problema para especificação de requisitos. Segundo eles, ambos os modelos podem ser utilizados para definir requisitos em diferentes níveis de abstração, os diagramas de problemas mostram o contexto do problema de cada

requisito, e modelos de objetivos mostram as ligações diretas entre o nível superior e nível inferior de requisitos.

Os requisitos de software são representados em um modelo de objetivo KAOS. Este modelo é um grafo *AND/OR*, onde os nós do grafo são os objetivos do sistema multiagente. As relações de refinamento *AND/OR* entre os nós mostram quais submetas precisam ser satisfeitas e quais propriedades e hipóteses de domínio precisam ser válidas para satisfazer um objetivo principal.

Já os diagramas de problema, foram sugeridos por Jackson (2000), eles mostram o futuro software, os requisitos a serem satisfeitos no ambiente, e a parte do ambiente que é relevante para satisfazer o requisito.

A proposta dos autores não identifica as crenças associadas aos objetivos, os planos necessários para atingir os objetivos nem quando os objetivos se tornam intenções. Dessa forma, a proposta não cobre integralmente o modelo BDI, ela suporta apenas identificação de objetivos.

Os autores Abushark et al. (2016), estendem a metodologia Prometheus proposta por Padgham e Winikoff (2002), para melhorar a compreensibilidade e capacidade de manutenção dos requisitos. O trabalho propõe a geração de diagramas de atividades *Unified Modelin Language* ou Linguagem de Modelagem Unificada (UML) a partir de modelos de requisitos existentes, tais como cenários e hierarquias de objetivos chamadas de árvores de objetivos. Esses diagramas complementam o processo do projeto de agente da metodologia Prometheus.

Atualmente, os requisitos do sistema são especificados por meio de cenários, objetivos e interfaces com o ambiente. Esses tipos de etapas incluem percepções, ações ou objetivos. Os objetivos podem ser decompostos em subobjetivos, usando uma árvore de objetivos. A combinação dos cenários junto com as árvores de objetivo representam os requisitos do sistema.

Os cenários apenas capturam uma sequência de etapas. Logo, as etapas que poderiam ser realizadas em paralelo só podem ser mostradas como uma sequência. Dessa forma, os autores propoem gerar diagramas de atividades equivalentes aos cenários e árvores de objetivos. O diagrama de atividades gerado representa e modela os fluxos de apenas um cenário.

Essa representação melhora a forma de analisar cenários, tendo em vista que, segundo Bratthall e Wohlin (2002) as notações gráficas tendem a melhor apoiar as tarefas que envolvem a compreensão da estrutura geral.

A proposta dos autores não permite a identificação de crenças associadas ao objetivo, não permite identificar quando os objetivos viram intenções, não identifica os planos necessários para atingir os objetivos e também não identifica os papéis de agentes. Já as representações gráficas são modeladas a partir de cenários propostos pela Prometheus sem adaptações. Outra limitação é que o diagrama de atividades é utilizado apenas com

recursos disponibilizados pela própria UML (UML. . . , 2021), sem adaptar para o contexto de sistemas multiagentes.

O estudo apresentado por Ribino et al. (2013), foca na modelagem de organizações BDI para análise de requisitos. Os autores propõem o uso de uma ontologia como modelo de análise representando a realidade do domínio do problema e a incompletude da especificação do problema.

As duas atividades propostas resultam em uma análise do modelo composto por uma ontologia e um modelo de objetivo. Já o modelo de objetivo é usado para descrever os objetivos das partes interessadas envolvidas no problema e as dependências entre eles. O autor utiliza a definição de objetivo descrito por Yu (1996), no qual o objetivo é descrito como uma condição ou estado do mundo que o ator deseja alcançar.

Na metodologia proposta pelos autores, a identificação do objetivo é a atividade preliminar da fase de modelagem de objetivo do agente, em que o analista torna explícito os objetivos estratégicos do sistema. A identificação dos objetivos do usuário é normalmente conduzida nas fases iniciais da análise de requisitos.

Os autores focam o estudo em métodos para identificar sistematicamente objetivos (*goals*) do domínio e de usuários. A principal contribuição do trabalho, segundo o autor, é que duas partes do processo de projeto são colocadas no início da fase de análise, para construir a ontologia do domínio do problema de forma útil para identificar objetivos.

A proposta apresentada pelos autores, permite a especificação de crenças e desejos, mas não permite a identificação de quando os objetivos se tornam intenções. Outra desvantagem do uso da metodologia proposta nesse trabalho é que não apresenta a subárea de validação da especificação de requisitos.

O estudo dos autores Argente, Botti e Julián (2009), propõe a GORMAS (*Guidelines for Organizational Multi-agent Systems* ou Diretrizes para Sistemas Multiagentes Organizacionais) é composto por quatro fases: I) análise de missão; II) análise de serviço; III) projeto organizacional; e IV) projeto dinâmico da organização.

A fase de análise de missão, descreve os objetivos globais do sistema, os serviços e produtos que o sistema fornece a outras entidades, as partes interessadas e as condições do ambiente. Tem como produto de trabalho um modelo de diagrama e três documentos de texto. Já a fase de análise de serviço, descreve o tipo de produtos e serviços, e as tarefas e objetivos relacionados à sua produção; os recursos e aplicativos necessários para oferecer a funcionalidade do sistema e as funções relacionadas com as partes interessadas.

Na fase de projeto organizacional, as dimensões organizacionais são definidas e são empregados para definir uma estrutura adequada para a organização. Já a fase de projeto dinâmico da organização define a comunicação entre agentes, processos em relação a funções e agentes, mecanismos para o controle do sistema (como normas) e um sistema de recompensa.

A fase de análise de missão produz quatro artefatos, o primeiro é um diagrama

comportamental, chamado de Modelo de Dimensão Funcional. Os outros três produtos de trabalho são textos estruturados, que são o documento de missão organizacional, documento de partes interessadas e documento de condições ambientais.

O modelo de dimensão funcional detalha a funcionalidade específica do sistema, com base nos serviços, tarefas e objetivos, bem como nas interações do sistema. O documento de missão organizacional é empregado para definir a missão de organização que será descrita. Define-se os resultados que o sistema irá fornecer, partes interessadas em manter um relacionamento com a organização e justificativa para projetar o sistema.

O documento de partes interessadas fornece a descrição das partes interessadas e o tipo de parte interessada de todas que foram identificadas no documento de missão organizacional. Fornece os objetivos que as partes interessadas seguem, seus produtos e serviços prestados e requeridos, os benefícios obtidos por eles e sua posição na organização.

E por fim, o documento de condições ambientais descreve cinco condições: a taxa de mudança, a complexidade, a incerteza, a receptividade e a diversidade de um ambiente. Utilizado para definir as condições ambientais nas quais a organização será inserida.

Na fase de análise de serviço, os serviços oferecidos pela organização aos seus clientes são especificados, bem como a forma como esses serviços se comportam e quais são as relações entre eles. Além disso, os objetivos associados aos serviços são detalhados. Esta fase gera quatro produtos de trabalho, sendo dois diagramas do modelo GORMAS e dois documentos de textos.

Modelo de dimensão estrutural descreve os componentes do sistema e seus relacionamentos. Nessa etapa os agentes e o relacionamento com as partes do software são identificados. Já o modelo de dimensão de ambiente representa o espaço de trabalho onde a unidade organizacional está localizada.

O documento de identificação de produto/serviço descreve o tipo de produção que o sistema terá e a tecnologia que os produtos usarão. Já o documento de descrição de produto/serviço descreve os produtos e serviços que a organização irá produzir. Os serviços são descritos por meio de sua funcionalidade, os papéis envolvidos sobre eles e seu perfil.

Observamos que essa metodologia cobre as subáreas de análise e especificação de requisitos. No entanto, não identifica as crenças associadas aos objetivos, não identifica as percepções sobre o ambiente e não identifica quando os objetivos se tornam intenção. Dessa forma, consideramos que ela suporta parcialmente o modelo BDI.

O estudo apresentado por Huiying e Zhi (2009), define um grupo de métodos de representação orientados a agentes com uso de símbolos gráficos e linguagem de modelagem de formalização. Os autores propõem um conjunto de estruturas de metamodelos para a engenharia de requisitos. O metamodelo estabelece figuras gráficas para representar requisitos não funcionais, atores, objetivos, recursos, tarefas e formas de comunicação entre agentes.

A proposta dos autores combina a representação gráfica com um método de modelagem formal. Este formalismo permite definir os atores, objetivos, requisitos não funcionais, tarefas, sequência de execução de atividades, e um gatilho que representa a ocasião em que o ator tenta atingir seu objetivo, ou seja, quando um objetivo vira intenção.

Observamos que o estudo apresentado pelos autores cobre as subáreas de análise e especificação de requisitos. No entanto, ela não permite identificar papéis de agentes, planos associados aos objetivos e também não permite identificar crenças. Dessa forma, classificamos este estudo como atendimento parcial do modelo BDI.

Os autores Bryl et al. (2008), apresentam no artigo a pesquisa desenvolvida por eles na área de especificação de requisitos de processos de negócios. É proposto uma integração de diferentes técnicas com o objetivo de conciliar a elicitação de requisitos com a especificação declarativa, prototipagem e análise dentro de uma única estrutura unificada.

A proposta dos autores utiliza a Tropos como base porque ela utiliza conceitos de agentes e objetivos desde o início das fases de desenvolvimento. Mas uma desvantagem do uso da Tropos para o contexto de processo de negócios é que ela não permite a modelagem de restrições temporais e de dados entre tarefas atribuídas aos agentes. Ou seja, horários de início, conclusão e prazos são elementos essenciais na descrição de modelo de processo de negócio e não são cobertos pela Tropos.

Assim como Tropos, a proposta dos autores é baseada em modelos que usam os conceitos de atores (ou seja, agente e funções), objetivos, tarefas, recursos e dependências sociais entre dois atores. Um ator é uma entidade ativa que tem objetivos estratégicos e executa ações para alcançá-los. Um objetivo representa um interesse estratégico de um ator. Uma tarefa representa um curso particular de ações que produzem um efeito desejado. Um recurso representa uma entidade física ou informativa sem intencionalidade. Uma dependência entre dois atores indica que um ator depende de outro para atingir algum objetivo, executar alguma tarefa ou entregar algum recurso.

A extensão de Tropos, proposta na B-Tropos, permite a especificação de informações temporais em objetivos e tarefas (planos). Cada objetivo ou tarefa pode ser descrito em termos de sua duração permitida, como tempo mínimo e tempo máximo. Assim como em Tropos, B-Tropos não permite a identificação de quando um objetivo se torna intenção. Também não permite identificar as crenças do agente. Dessa forma, classificamos este estudo como atendimento parcial ao modelo BDI.

Também observamos que o estudo cobre as subáreas de elicitação, análise e especificação de requisitos. No entanto, não cobre a subárea de validação, tão importante para a garantia da qualidade de sistemas de software.

Segundo os autores Ranjan e Misra (2006a), a maioria das abordagens de engenharia de requisitos analisa os requisitos funcionais e não funcionais em um único módulo. Mas como cada requisito precisa de várias interações com diferentes usuários do sistema,

fica difícil analisar os requisitos funcionais e não funcionais em um único módulo. A maioria das abordagens não integra os atributos de qualidade com os requisitos funcionais na fase de coleta e análise de requisitos.

A metodologia proposta utiliza o modelo de casos de uso, modelo de interação e modelo de protocolo da metodologia ROADMAP (JUAN; PEARCE; STERLING, 2002). Já os demais modelos são utilizados conforme apresentado no trabalho anterior de Ranjan e Misra (2006b).

O estudo de Ranjan e Misra (2006a) propõe uma metodologia de análise de requisitos em que a fase de análise é dividida em análise centrada no usuário e análise centrada no sistema. Este refinamento é obtido identificando e separando todos os requisitos em três tipos (I) requisitos funcionais; (II) requisitos não funcionais; e (III) requisitos de corte transversal.

Essa metodologia propõe o uso de modelo de casos de uso, modelo de interação e modelo de protocolo a partir da metodologia ROADMAP (JUAN; PEARCE; STERLING, 2002). Os outros modelos são usados conforme apresentado no trabalho anterior de Ranjan e Misra (2006b).

Na fase de análise centrada no usuário é possível identificar os requisitos transversais com base nos requisitos funcionais e não funcionais identificados. Os requisitos transversais representam os requisitos não funcionais que afetam mais de um requisito funcional.

Já na fase de análise centrada no sistema são utilizados os seguintes modelos: I) Modelo de Objetivo Centrado no Sistema; II) Modelo de Papéis; III) Modelo de Interação; e IV) Modelo de Protocolo. No Modelo de Objetivo Centrado no Sistema os requisitos funcionais e não funcionais são analisados do ponto de vista do sistema com base em suas entradas, saídas e controles. Ao refinar os objetivos é possível identificar os agentes associados e suas propriedades. O modelo de papel descreve as propriedades do papel de agente contendo a identificação e descrição do papel, identificação de tarefas associadas e restrições envolvidas.

Em relação ao modelo BDI, essa metodologia, como ROAD permite apenas a identificação de objetivos, não identifica crenças associadas aos objetivos, também não identifica os planos necessários para atingir os objetivos nem quando os objetivos se tornam intenções. Também observamos que o estudo cobre as subáreas de elicitação, análise e especificação de requisitos. Dessa forma, não cobre a subárea de validação de requisitos.

Segundo os autores Cossentino et al. (2010), ASPECS é um processo de software orientado a agente para engenharia de sistemas complexos. Ela permite a modelagem de um sistema com diferentes níveis de detalhes usando um conjunto de métodos de refinamento. O processo possui três fases: I) Análise de Requisitos do Sistema; II) Projeto de Sociedade de Agente; e III) Implementação e Implantação. No entanto, como o foco de nosso estudo é na engenharia de requisitos, destacamos apenas as fases de análise e

especificação cobertas no ASPECS.

A fase de Análise de Requisitos do Sistema visa identificar uma hierarquia de organizações, cujo comportamento global pode cumprir os requisitos do sistema sob a perspectiva escolhida. Começa com uma atividade de Descrição de Requisitos de Domínio (DRD) onde os requisitos são identificados usando técnicas clássicas, como casos de uso. A UML (UML. . . , 2021) foi estendida para atender às características específicas de agentes, para isso foi utilizado perfis específicos (estereótipos). Os estereótipos (*Stereotype*) são um mecanismo de extensibilidade da UML. Eles dão mais poder à UML, permitindo classificar elementos com algo em comum.

O conhecimento do domínio e o vocabulário associado ao domínio do problema são coletados e descritos na atividade de Descrição da Ontologia do Problema (POD). Na sequência os requisitos são associados às organizações recém-definidas. Cada organização será, portanto, responsável por exibir um comportamento que preenche os requisitos pelos quais é responsável.

Para identificar os papéis de agentes, o processo proposto pelos autores utiliza o conceito de capacidade. Uma capacidade é uma descrição abstrata de uma competência de uma papel. Cada função requer certas habilidades para definir seu comportamento, e essas habilidades são modeladas por meio de uma capacidade.

A fase de Análise de Requisitos gera oito produtos de trabalho. O primeiro é a Descrição de Requisitos do Domínio, que é composto por diagramas de caso de uso UML (UML. . . , 2021) padrão, ou seja, sem adaptações, documentação textual de casos de uso e descrição textual de requisitos não funcionais. O próximo documento é o Glossário que corresponde a um documento de texto estruturado que define os termos usados nas fases de desenvolvimento.

Outro documento é a Descrição de Ontologia do Problema, que descreve conceitos de termos (categorias do domínio), predicados (afirmações sobre propriedades do conceito), ações (realizadas no domínio, afetando o status dos conceitos), e seus relacionamentos. Nesse documento é utilizado um perfil para diagramas de classes UML (UML. . . , 2021) para descrever a ontologia.

Já a Identificação de Papéis e Interações, contém um diagrama de classes onde cada papel é representado por uma classe estereotipada e as interações são representadas por associações entre os papéis. São utilizados pacotes para representar as organizações a que os papéis pertencem.

O produto de trabalho "Descrição de Cenários", contém os cenários descritos utilizando-se diagramas de sequência estereotipados para representação dos papéis. Já o Plano de Papéis, utiliza diagramas de atividades, onde as raias são usadas para particionar atividades de funções diferentes e uma raia separada é utilizada para representar eventos externos.

E por fim, o produto de trabalho Identificação de Capacidades, utiliza um diagrama

de classes estereotipado onde as capacidades são relacionadas às funções que os requerem por meio de associações.

O ASPECS permite apenas a identificação de objetivos a serem atingidos, dessa forma, atende parcialmente o modelo BDI, pois o processo não identifica crenças e intenções. Também não permite identificar quando os objetivos viram intenções. Também observamos que o estudo cobre apenas as subáreas de análise e especificação.

Os autores Bresciani e Donzelli (2003), apresentam um *Framework* para Engenharia de Requisitos (REF - *Requirement Engineering Framework*), que tem como objetivo fornecer aos analistas e às partes interessadas uma ferramenta poderosa para capturar necessidades organizacionais de alto nível e transformá-las em requisitos de sistema. Apresenta uma estrutura baseada em agentes e objetivos que permite às partes interessadas não técnicas elicitar requisitos.

Os agentes são usados para modelar a organização, já o contexto organizacional é modelado como uma rede de agentes interagindo, colaborando ou conflitando uns com os outros para atingir os objetivos individuais e organizacionais.

O REF é composto por três atividades: I) Modelagem Organizacional; II) Modelagem de Objetivos Difíceis; e III) Modelagem de Objetivos Suaves. A Modelagem Organizacional tem como propósito analisar o contexto organizacional e identificar os agentes e objetivos. Os objetivos identificados são refinados e a interação com os agentes é modelada por meio de objetivos difíceis (*Hard-goal*) e objetivos suaves (*soft-goal*).

A Modelagem de Objetivos Difíceis, procura determinar como um agente pode alcançar um objetivo recebido. Para isso os objetivos difíceis são decompostos em objetivos subordinados, dessa forma é compreendido como o agente pensa para atingir o objetivo, em termos de objetivos e tarefas.

Por fim, a atividade de Modelagem de Objetivos Suaves visa produzir definições da realização de um objetivo suave que são atribuídos ao agente que originou a intenção de atingir os objetivos. Os objetivos suaves correspondem aos requisitos não funcionais (SUTCLIFFE, 2001).

O estudo apresentado pelos autores permite identificar os objetivos e intenções, dessa forma, cobre parcialmente o modelo BDI. A especificação de requisitos é representada com a elaboração dos modelos Organizacional, de Objetivos Difíceis e de Objetivos Suaves. Este modelo utiliza uma representação de diagramas inspirados em i* proposto por Yu (1996). A proposta apresentada é do mesmo autor de Tropos (BRESCIANI et al., 2004), portanto pode ser integrado ou estendido para metodologia Tropos.

O estudo de Sutcliffe (2001), propõe um método para analisar requisitos para sistemas complexos baseado nos modelos i*. O modelos i*, propostos por Yu (1996), fornecem semântica para descrever agentes, objetivos, tarefas e as dependências entre eles. Mas esses modelos não permitem modelar como a comunicação é integrada com a ação. Com isso, a proposta do autor tem como objetivo suprir esta carência, visto

que é uma preocupação fundamental para a compreensão dos requisitos para sistemas de computação colaborativa.

A proposta é compreendida por três estágios, semelhantes a fases, a saber: I) Modelagem de Sistema e Casos de Uso; II) Análise de Tarefa e Comunicação; e III) Análise Dinâmica de Cenário. O primeiro estágio se refere à Modelagem de Sistema e Casos de Uso, que utiliza casos de uso para especificar as interações entre agentes e mapear as dependências de fluxos de eventos entre usuários e agentes do sistema. A modelagem do sistema é realizada utilizando-se uma versão extendida do i*.

Já no estágio de Análise de Comunicação, os fluxos de eventos de comunicação entre os agentes são classificados usando tipos de atos de discurso. Os atos do discurso são especificados com diferentes graus de prioridade e obrigação que estabelece a obrigatoriedade de executar as ações pelo receptor da comunicação. Nessa etapa o padrão da mensagem e os fluxos de entrada/saída para cada agente são investigados para determinar a capacidade do agente de responder às demandas. Estas informações são especificadas em uma tabela de comunicação.

Na Análise de Tarefas, as tarefas são analisadas e especificadas em uma tabela. Essa tabela contém as tarefas e ponderações considerando estimativas de complexidade e habilidade necessária para executar a tarefa. Tarefas simples recebem pesos baixos, enquanto tarefas complexas baseadas em conhecimento que requerem raciocínio recebem pesos maiores.

Por fim, no estágio de Análise Dinâmica o modelo do sistema é executado em um ou mais cenários. Cada evento de entrada de um agente ou de outras fontes no ambiente do sistema indica um requisito para cada resposta do sistema. As implicações para cada agente são avaliadas contando as tarefas que têm de realizar e as mensagens enviadas ou recebidas para processar o evento. Isso produz uma análise de carga de trabalho cognitiva para cada agente em termos de comunicação e tarefas.

A proposta do autor permite identificar objetivos associados aos agentes, mas não permite identificar crenças, intenções e planos. Dessa forma, classificamos como atendimento parcial do modelo BDI. Também observamos que cobre apenas as subáreas de análise e especificação, deste modo, a subárea de validação é negligenciada neste estudo.

A metodologia proposta por Caire et al. (2001), chamada de MESSAGE, integra alguns conceitos básicos relacionados ao agente, como organização, papel, objetivo e tarefa. A notação MESSAGE estende a UML (UML..., 2021) com conceitos de nível de conhecimento do agente e fornece notações gráficas para visualizá-los. Os diagramas propostos estendem a classe UML (UML..., 2021) e diagramas de atividades.

O estudo proposto pelos autores, adota o ciclo de processo do processo RUP (KRUCHTEN, 2000). No entanto, o MESSAGE tem foco nas fases de Análise e Projeto. Discutiremos apenas a fase de Análise, tendo em vista que nosso foco é na engenharia de requisitos. A metodologia propõe cinco modelos, a saber:

- Organizacional: na visualização organizacional são especificados todos os elementos necessários para o desenvolvimento de um SMA, ou seja, os agentes, organização, papéis, recursos (como banco de dados e serviços de aplicativos), sistema, ambiente e seus relacionamentos;
- Objetivo/Tarefas: especifica em detalhes os objetivos que os agentes/papéis perseguem e as tarefas que realizam para alcançar os objetivos;
- Agente/Papel: especifica individualmente os agentes e papéis. Essa visão descreve suas características, como por exemplo quais objetivos os agentes/papéis são responsáveis, quais eventos eles precisam sentir, quais recursos eles controlam, as regras de comportamento necessárias, dentre outras.
- Interação: este modelo especifica a interação entre Agentes/Papéis, o iniciador, os colaboradores, o motivador (geralmente um objetivo pelo qual o iniciador é responsável), as informações relevantes fornecidas/alcançadas por cada participante, os eventos que desencadeiam a interação e outros efeitos da interação. Por exemplo, um agente se tornando responsável por um novo objetivo; e
- Domínio: especifica os conceitos e relações específicos do domínio que são relevantes para o sistema em desenvolvimento.

A metodologia MESSAGE, estende a UML (UML. . . , 2021) utilizando o diagrama de classes e o diagrama de atividades para especificação dos cinco modelos. Essa metodologia cobre apenas a subárea de Análise da Engenharia de Requisitos. No entanto, observamos que essa metodologia permite a especificação de requisitos com base nos cinco modelos presentes em sua estrutura, dessa forma entendemos que cobre a subárea de Especificação de requisitos. Também observamos que permite identificar apenas objetivos dos agentes como característica interna, deste modo, classificamos essa metodologia como suporte parcial do modelo BDI.

A metodologia proposta por DeLoach e Garcia-Ojeda (2014) denominada de O-MaSE, apresenta modelos e tarefas para a execução de ER, mas não define um ciclo de vida ou obrigatoriamente a utilização desses modelos.

O-MaSE consiste em três estruturas: I) um processo; II) um conjunto de fragmentos de método; e III) um conjunto de diretrizes. O processo O-MaSE define os conceitos-chave necessários para projetar e implementar sistemas multiagentes. Os fragmentos de método são tarefas que são executadas para produzir um conjunto de produtos de trabalho, que pode incluir modelos, documentação ou código. As diretrizes definem como os fragmentos do método estão relacionados entre si.

O-MaSE foi projetado para ser utilizado em um conjunto de fases que melhor se adeque ao produto de software que está sendo desenvolvido, essa metodologia não possui uma sequência pré-definida de atividades. De qualquer modo, o processo geral

do O-MaSE é composto pelas seguintes fases: I) Análise de Requisitos; II) Projeto; e III) Implementação. A fase que dá suporte a Engenharia de Requisitos é de Análise de Requisitos e envolve as seguintes atividades:

- Levantamento de requisitos: envolve a tarefa de especificação de requisitos e o produto de trabalho é a especificação de requisitos do software que está sendo desenvolvido;

- Análise de requisitos: envolve as tarefas de modelagem de objetivos, refinamento de objetivos e modelagem do domínio. Os produtos de trabalho dessa fase são o Modelo de objetivos e Modelo de Domínio; e

- Análise da solução: envolve as tarefas de modelagem organizacional, interfaces, modelagem de papéis, definição de papéis e definição de objetivos de papéis. Os produtos de trabalho são o Modelo organizacional, Modelo de papéis, Documento de descrição de papéis e Modelo de objetivos de papéis.

A metodologia O-MaSE cobre apenas as subáreas de Elicitação, Análise e Especificação de requisitos. Ela negligencia a fase de validação de requisitos que é tão importante para a garantia da qualidade do software em desenvolvimento. Além disso, verificamos que permite a representação de objetivos, dessa forma, suporta parcialmente o modelo BDI.

A metodologia Gaia, apresentada pelos autores Cossentino et al. (2014b) foca no uso de abstrações organizacionais para conduzir a análise e projeto de SMAs. Gaia modela tanto os aspectos macro (sociais) quanto os aspectos micro (internos do agente) de um SMA. Atualmente ela propõe três fases principais, ou seja, a Análise, o Projeto Arquitetônico e o Projeto Detalhado. Segundo os autores, Gaia não lida diretamente com a fase de requisitos. Discutiremos apenas a fase de análise pois as fases de projeto não fazem parte do escopo de nosso estudo.

O processo de Gaia é principalmente dedicado a representar um SMA como uma organização social, e adota organizações, agentes, papéis, protocolos, regras organizacionais e ambiente como seus blocos de construção básicos. Um agente Gaia é uma entidade que desempenha um ou mais papéis.

A fase de Análise da Gaia identifica as suborganizações do SMA com seus respectivos objetivos. Em seguida, os papéis de agente são especificados no "Modelo de Papel". A interação entre os papéis são especificados no "Modelo de Interação Preliminar". Já o "Modelo Ambiental" especifica o ambiente no qual os papéis de agentes interagem para atingir seus objetivos.

Além dos modelos produzidos na fase de análise, a metodologia Gaia não apresenta um modelo de documento de especificação de requisitos com uma estrutura formalizada.

Conforme apresentado na metodologia a especificação de requisitos deve ser documentada em um texto livre.

Observamos que Gaia não apresenta uma engenharia de requisitos bem estruturada, ela tem um foco maior nas fases de projeto. Dessa forma, a especificação de requisitos é superficial e não apresenta fase de validação dos modelos criados. No entanto, como ela permite a identificação de objetivos, classificamos como atendimento parcial do modelo BDI.

Os autores Ronald et al. (2012), apresentam um modelo social baseado em agentes com objetivo de explorar os efeitos do espaço social nas tomadas de decisões de pessoas. O estudo propõe um ambiente baseado em agente para representar decisões por pessoas no mundo real. Um processo é utilizado como base para desenvolvimento da proposta dos autores, este processo tem como objetivo descrever um sistema multiagente e seus aspectos, tais como: atores, ambiente, interação e organização.

É proposto uma definição de perspectiva externa e interna do agente. A perspectiva externa se considera o que é observável no ambiente. Tem como entrada as percepções fornecidas pelo ambiente. Já a perspectiva interna, considera as representações mentais sobre o ambiente, de forma geral, qualquer coisa que seja importante de interpretação.

No entanto, os modelos propostos no estudo dos autores, leva em consideração aspectos específicos de agentes atuantes em redes sociais. Dessa forma, é limitado para este tipo de sistema e não pode ser aplicado em outros sistemas multiagentes com características distintas. Quanto à cobertura do modelo BDI observamos que o estudo cobre parcialmente este modelo, tendo em vista que permite a identificação apenas de objetivos (*Desire*).

Também observamos que este estudo cobre apenas as subáreas de análise e especificação de requisitos, dessa forma, não cobre as subáreas de elicitação e validação, esta última tão importante para garantia da qualidade.

O estudo apresentado pelos autores Wang et al. (2013) propõe uma linguagem de padrão chamada de PLANT (*Pattern LANguage for Transforming scenarios* ou Padrão de Linguagem para Transformação de Cenários). Cada padrão transforma parte dos cenários em um modelo. O padrão “Estabelecer a linha da história” se concentra no aspecto do processo do cenário e especifica sua sequência de ação em um Modelo Orientado ao Processo. Já o padrão “Elaborar Coisas que Mudam” enfoca o aspecto de objeto do cenário e representa a dinâmica dos objetos em um Modelo de Transformação de Objeto.

O padrão “Identificar agentes e suas interações” enfoca o aspecto do agente no cenário e expressa a colaboração do agente usando um “Modelo Orientado a Agentes” . E por fim, o padrão “Desenvolver o objetivo e seus Subobjetivos” se concentra no aspecto do objetivo e tem como produto de trabalho o “Modelo Orientado a Objetivo”. Com base nos modelos criados com os padrões citados anteriormente, observamos que a proposta dos autores permite a especificação apenas de objetivos. O que nos leva a conclusão que

cobre parcialmente o modelo BDI.

Outra limitação da proposta é que cobre apenas as subáreas de Análise e Especificação de requisitos, não apresenta as subáreas de Elicitação e Validação de requisitos.

Os autores Cao et al. (2004) propoem um *framework* orientado a objetivos para modelagem de requisitos iniciais. Dessa forma, o i* estendido foi usado para construir modelos visuais e uma especificação formal complementa a modelagem visual para definir e refinar os requisitos. Essa proposta utiliza o conceito de Modelagem Integrativa, que é uma metodologia que tenta integrar especificações formais com uma notação informal diagramática. A primeira etapa, que corresponde à Análise de requisitos funcionais, tem como preocupação a compreensão dos objetivos do sistema e estudo do ambiente organizacional.

Nessa etapa é utilizado o i* estendido, que oferece identificação de ator Modelagem Integrativa. A primeira etapa corresponde à "Modelagem Visual", nessa etapa os autores não restringem a notação utilizada, deixando em aberto a possibilidade de se adotar diagramas de casos de uso ou *framework* i*. Por fim, a etapa de "Modelagem Interativa", tem como objetivo integrar as especificações formais e notações informais que usam diagramas.

A proposta dos autores permite apenas a identificação de objetivos, não permitindo a identificação de crenças e intenções. Também não permite identificar quando os objetivos viram intenções. Outra desvantagem do uso da proposta apresentada pelos autores é que não cobre as etapas de elicitação nem a subárea de validação de requisitos, esta tão importante ra garantia da qualidade na engenharia de requisitos.

Os autores Haumer et al. (1999) descrevem uma solução baseada em cenário para o desenvolvimento de SMAs. Envolve a integração de cinco etapas, a saber:

- A captura persistente de contexto na forma do mundo real através de cenas capturadas em multimídia;
- Modelagem formal orientada a agentes com uma semântica que permite animação interativa distribuída;
- Diagramas de rastreamento de mensagem como um meio para troca casos de teste de animação e rastreios;
- Um modelo de objetivo para controlar e registrar o processo de ER; e
- Um ambiente de ferramenta integrado ao processo para garantir a orientação do método e rastreabilidade com o mínimo esforço possível.

Um dos problemas difíceis na gestão de cenários é a escolha de uma representação adequada. No entanto, os cenários de texto também têm suas limitações. A validação de modelos parte da necessidade de manter a rastreabilidade dos artefatos produzidos

na engenharia de requisitos. O argumento principal dos autores é que as soluções para o problema de visualização e para o problema de rastreabilidade devem ser combinados para alcançar uma solução eficaz para os problemas de mudanças contínuas de requisitos.

Mesmo dentro de um único Processo de RE, tal integração poderia melhorar significativamente o ciclo entre a elicitação de requisitos com base em uma análise do estado em que se encontram e a definição e validação do futuro modelos de requisitos de sistema. A proposta aborda exatamente o tipo de aplicativo em que cenários por si só são insuficientes e, portanto, colocam ênfase na captura de multimídia orientada a objetivo e animação do estado atual e futuros modelos. Novos componentes do modelo podem ser extraídos das gravações , ou modelos de estado atual existentes podem ser validados contra eles.

Embora a metodologia suporta a subárea de validação, ela não apresenta a forma como os requisitos serão validados, também não apresenta a técnica que será utilizada para validação dos requisitos. O estudo suporta parcialmente características necessárias para o modelo BDI.

O autor Banach (2010), propõe em seu estudo um "*framework*" denominado de KAOS o qual é utilizado para engenharia de requisitos direcionada a objetivos, baseada na decomposição e refinamento de objetivos.

Em geral, uma especificação KAOS é uma coleção de três modelos, a saber:

- Modelo de metas: reúne um conjunto de metas cujo objetivo é representar a melhor forma de se resolver um problema. Os obstáculos se relacionam diretamente com metas, submetas, requisitos e expectativas por meio de arestas direcionadas;
- Modelo de objetos: modelo UML (UML. . . , 2021) que pode ser derivado de especificações formais de metas, uma vez que se referem a objetos e suas propriedades;
- Modelo de operação: define os vários serviços a serem prestados por agentes de software;
- Modelo de responsabilidade: descreve todos os diagramas de responsabilidade de um sistema. Esses diagramas descrevem para cada agente, os requisitos e expectativas dos quais ele é responsável ou que tenha sido atribuído a ele.

Observamos que este estudo cobre apenas as subáreas de análise e especificação de requisitos. Dessa forma, não possui em sua estrutura as subáreas de elicitação e validação. Também observamos que KAOS identifica apenas os objetivos, com isso, classificamos como atendimento parcial ao modelo BDI.

Estudos que suportam totalmente o modelo BDI:

Apresentamos na Tabela 7, quais artefatos, modelos ou documentos são utilizados para especificação de requisitos e as lacunas de cada estudo que suporta totalmente o modelo BDI.

Tabela 7 – Especificação de requisitos e lacunas de cada estudo que suporta totalmente o modelo BDI.

Metodologia	Como Especificam Requisitos	Lacunas
Cysneiros e Yu	• Modelo Orientado ao Agente. • Modelo de Dependência Estratégica. • Modelo de Primeiro Corte. • Modelo de Relacionamento Estratégico.	• Não apresenta nenhuma técnica para validação dos requisitos. • Não permite especificar o padrão de comunicação. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos.
PRACTIONIST	• Modelo de Objetivos	• Não cobre a subárea de Validação. • Não permite especificar o padrão de comunicação. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos.
Tropos	• Modelo de Ator de Requisitos Iniciais. • Modelo de Objetivo de Requisitos Iniciais. • Diagrama de Ator de Requisitos Tardios. • Diagrama de Objetivo de Requisitos Tardios. • Tabela de Capacidade de Requisitos Tardios.	• Não apresenta nenhuma técnica para validação dos requisitos. • Não permite especificar o padrão de comunicação. • Não permite especificar a comunicação entre agentes ou agentes e usuários externos. • Não fornece recursos para identificação e modelagem de perpepções associadas aos agentes.
MAS-CommonKADS	• Modelo de Agente. • Modelo de Tarefa. • Modelo de Perícia. • Modelo de Organização. • Modelo de Coordenação. • Modelo de Comunicação.	• Não cobre as subáreas de Elicitação e Validação. • Não permite especificar o padrão de comunicação. • Não permite especificar a comunicação entre agentes ou usuários externos. • Não fornece recursos para identificação e modelagem de percepções.

Fonte: Autor.

Conforme pode ser observado na Tabela 7, no total 4 estudos que cobrem a subárea de especificação de requisitos suportam totalmente o modelo BDI. A seguir discutiremos em mais detalhes cada um desses estudos.

O estudo de Cysneiros e Yu (2002), apresenta uma metodologia baseada em conceitos de agentes no nível de modelagem de requisitos. Primeiramente os atores são identificados, e isto serve como um eixo central para elicitação e análise de requisitos. Os autores complementam e estendem a estrutura de modelagem do i* proposta por Yu (1996).

Uma Léxicon é usada para ajudar a construir um modelo orientado ao agente da estrutura social do domínio. Um modelo de Dependência Estratégica é então construído, para fornecer uma visão geral dos processos existentes e sistemas relacionados. À medida que a elicitação prossegue, o modelo de primeiro corte fornece uma estrutura para as partes interessadas expressarem suas preocupações e aspirações, criticar os processos atuais e os arranjos organizacionais e articular possíveis novos arranjos. As intenções dos agentes e suas relações internas são expressas no modelo de Relacionamento Estratégico.

As tarefas que são identificadas no processo, referem-se às diferentes maneiras em que um objetivo pode ser alcançado, ou seja, representam os planos. As desvantagens dessa metodologia é que não permite a identificação de quando os objetivos viram intenção e a subárea de validação não é apresentada com detalhes de como os requisitos devem ser validados, apenas é mencionado que os modelos criados devem ser validados sem apresentar como isso deve ser feito.

Os autores Morreale et al. (2006), definem a proposta do trabalho como um *framework* PRACTIONIST. Ele oferece suporte aos programadores para o desenvolvimento de agentes BDI. Esses agentes são especificados com uma representação de seus estados internos e o ambiente externo. Também são representados considerando a capacidade de planejar suas atividades a fim de perseguir alguns objetivos.

No PRACTIONIST, um objetivo é considerado como uma abstração de análise, projeto e implementação. Em outras palavras, agentes PRACTIONIST podem ser programados em termos de objetivos, que então estarão relacionados a desejos ou intenções de acordo se algumas condições específicas são satisfeitas ou não.

No *framework* PRACTIONIST, os planos representam um importante contêiner no qual os desenvolvedores definem os comportamentos e estratégias reais dos agentes. Os planos PRACTIONIST têm um conjunto de slots, que são usados pelos agentes durante o raciocínio meio-fim e o real execução de suas atividades. Alguns desses slots são: o evento desencadeador, que define o evento (ou seja, objetivos, percepções e atualização de crenças) que cada plano deve tratar; o contexto, um conjunto de condições que devem ser consideradas verdadeiras antes de executar o plano; o corpo, que inclui os atos que o agente realiza durante a execução do plano.

Os agentes podem executar vários atos como por exemplo enviar mensagens, realizar alguma ação, modificar crenças e assim por diante. Dessa forma, o estudo apresentado pelos autores cobre identificação de crenças, desejos e intenções, o que o torna uma abordagem com total cobertura ao modelo BDI.

O PRACTIONIST não apresenta fases concretas para o processo, porém ao examinar a classificação e modelagem dos requisitos, podemos identificar as subáreas de análise e especificação de requisitos na engenharia de requisitos do PRACTIONIST.

No entanto, uma desvantagem do uso deste estudo é que não cobre a subárea de validação de requisitos que é tão importante para a garantia da qualidade Além disso, não permite especificar quando os objetivos viram intenções.

Os autores Mylopoulos, Castro e Kolp (2013), apresentam um estudo o qual os estilos organizacionais inspirados na teoria da organização e alianças estratégicas são usados para projetar a arquitetura geral do sistema. A metodologia apresenta fases de projeto, no entanto, elas não serão discutidas porque fogem do escopo de nosso estudo.

As práticas atuais do uso de padrões em desenvolvimento de sistemas multiagentes torna o núcleo do aplicativo altamente acoplado aos padrões utilizados, dessa forma reduz

as oportunidades de reutilização.

Para resolver essa limitação, os autores propuseram uma técnica de descrição de padrões de projeto orientada a agentes, chamada Especificações de Padrão de Agente Implementação. Quando Tropos foi estendida, denominada de I-Tropos, ela incluiu disciplinas essenciais, tais como, Modelagem Organizacional, Requisitos Engenharia, Projeto de Arquitetura, Projeto Detalhado, Implementação, Teste e Implantação, mas também oferece suporte a disciplinas para lidar com Gerenciamento de Risco, Gestão do Tempo, Gestão da Qualidade e Gestão de Processos de Software.

Uma especificação formal de Tropos descreve os objetos relevantes dos domínios modelados e seus relacionamentos. A camada externa é semelhante a uma declaração de classe, uma vez que define a estrutura das instâncias junto com seus atributos.

O estudo apresentado pelos autores permite a identificação de características necessárias para o modelo BDI. Dessa forma, este estudo suporta totalmente o modelo BDI. Também observamos que cobre as subáreas de elicitação, análise e especificação de requisitos. No entanto, não cobre a subárea de validação de requisitos que é tão importante para a garantia da qualidade.

Os autores Iglesias et al. (1998), propõem a metodologia MAS-CommonKADS que estende a metodologia CommonKADS proposta inicialmente por Schreiber et al. (1994). Na fase de análise os requisitos serão especificados por meio de seis modelos, a saber:

- Modelo de agente (AM): especifica as características do agente, como por exemplo capacidades de raciocínio, habilidades (sensores/efetores), serviços, grupos de agentes e hierarquias (ambos modelados no modelo de organização).

- Modelo de tarefa (TM): descreve as tarefas que os agentes podem realizar, envolvendo os objetivos, métodos de resolução de problemas, entre outros.

- Modelo de perícia (EM): descreve o conhecimento necessário de cada agente para alcançar seus objetivos.

- Modelo de organização (OM): descreve a organização para a qual o SMA está sendo introduzido e a organização social da sociedade de agente.

- Modelo de coordenação (CoM): descreve a comunicação entre os agentes, suas interações, protocolos e recursos necessários.

- Modelo de comunicação (CM): detalha as interações homem-agente de software, e os fatores humanos para desenvolver essas interfaces de usuário.

Os agentes podem ser identificados com alguns dos modelos definidos no estudo, ou uma combinação deles. Na fase de análise os casos de uso são criados para delimitar os agentes externos do sistema. A modelagem dos agentes utiliza casos de uso internos

e cartões CRC (Class Responsibility Collaboration). A notação UML (UML..., 2021) foi estendida para diferenciar os agentes humanos (com cabeça redonda) dos agentes de software (com cabeça quadrada). Os cartões CRC permitem a descrição dos objetivos, dos planos para atingir essas metas, dos conhecimentos necessários para realizar os planos, dos agentes envolvidos nos planos e serviços usados na colaboração para atingir os objetivos.

A metodologia apresentada pelos autores, cobre apenas as subáreas de análise e especificação de requisitos. As subáreas de elicitação e validação não estão presentes na estrutura do MAS-CommonKADS. Por outro lado, em relação ao modelo BDI, a metodologia não menciona esse modelo diretamente, porém, na descrição dos modelos utilizados na metodologia várias vezes é mencionado o uso de crenças, desejos e intenções, o que nos faz acreditar que a metodologia apóia o modelo BDI.

Questão de pesquisa 5: Quais dessas metodologias suportam a subárea de validação de requisitos e quais técnicas são utilizadas?

Apresentamos na Tabela 8 os estudos retornados em nossa RSL que suportam a subárea de validação e sua cobertura em relação ao modelo BDI.

Tabela 8 – Metodologias/Processos que cobrem a subárea de validação e o suporte ao modelo BDI.

Metodologia	**Validação**	**Crença**	**Desejo (Objetivo)**	**Intenção**	**Não suporta BDI**
Cysneiros (CYSNEIROS; YU, 2002)	✓	✓	✓	✓	
CREWS-EVE (HAUMER et al., 1999)	✓		✓		
ADELFE (BONJEAN et al., 2014)	✓				✓

Fonte: Autor.

Conforme pode ser observado na Tabela 8 apenas 3 estudos suportam a subárea de validação de requisitos. Com base nisso, podemos concluir que grande parte dos estudos negligenciam esta importante área da engenharia de requisitos. Dessa forma, se configura como uma lacuna que pode ser explorada por novos estudos.

Entre esses estudos, observamos que a metodologia ADELFE (BONJEAN et al., 2014) é a única que não suporta nenhuma característica necessária ao modelo BDI. Também observamos que essa metodologia não apresenta nenhuma técnica para verificar ou validar os requisitos.

Já o estudo apresentado por Haumer et al. (1999) permite a identificação apenas de objetivos (Desejos), dessa forma, cobre parcialmente o modelo BDI. Assim como na metodologia ADELFE, este estudo não apresenta nenhuma técnica para verificação ou validação de requisitos em SMA.

Por outro lado, o estudo de Cysneiros e Yu (2002) é o único que suporta a subárea de validação e cobre integralmente o modelo BDI. No entanto, este estudo não apresenta um processo de validação que possa ser seguido e replicado para validação de requisitos em SMAs. Além disso, não apresenta nenhuma técnica para verificar ou validar os requisitos,

apenas menciona que os artefatos produzidos devem ser validados, sem apresentar de que forma isso deve ser realizado.

Questão de Pesquisa 6: Quais são as lacunas existentes nas metodologias que suportam ER para SMA?

Percebemos que apenas três estudos abrangem a subárea de validação em seu ciclo de ER. Isso demonstra que a maioria das metodologias não se preocupa com essa fase tão importante para a qualidade dos sistemas.

Notamos também que apenas um estudo abrange as quatro subáreas da ER em seu ciclo (BONJEAN et al., 2014). Por outro lado, esse estudo não suporta o modelo BDI, o que demonstra uma lacuna e a necessidade da proposição de uma metodologia contendo todas as subáreas da engenharia de requisitos e que suporte o modelo BDI.

Em relação à cobertura do modelo de BDI, entendemos que o apoio a apenas 8 estudos de um total de 54 é um número baixo. Além disso, apenas duas metodologias têm como foco cobrir este modelo ((JO; EINHORN, 2005), (RIBINO et al., 2013)) e nenhuma delas cobre a elicitação e validação, o que evidencia uma lacuna na área de ER para SMA.

Outro ponto que poderíamos identificar como negligência é que, entre as metodologias que suportam o BDI, apenas a metodologia Tropos (MYLOPOULOS; CASTRO; KOLP, 2013) cobre a elicitação de requisitos e apenas a metodologia proposta por Cysneiros (CYSNEIROS; YU, 2002) inclui a validação de requisitos. Isso demonstra que a maioria das metodologias que suportam o BDI tem como foco a análise e especificação de requisitos e que, além dessas áreas, há espaço a ser explorado nas áreas de elicitação e validação.

3.1.9 Ameaças à Validade

Durante o planejamento e execução dessa revisão, alguns fatores foram caracterizados como ameaças à validade da pesquisa. Nas seções à seguir apresentamos as ameaças potenciais e ações realizadas para orientar a interpretação deste trabalho.

3.1.9.1 Validade de Construção

A confiabilidade da *String* de pesquisa definida para selecionar obras relevantes pode ser uma ameaça para a construção. Para minimizar essa ameaça, a *String* foi calibrada com a execução de diversos testes e o especialista da área foi consultado sobre os termos mais utilizados.

3.1.9.2 Validade Interna

Uma possível ameaça pode ter surgido da interpretação individual de cada pesquisador, o que pode ter levado à exclusão de estudos relevantes. Para minimizar essa

ameaça, o protocolo dessa revisão foi rigorosamente seguido, considerando principalmente os critérios de inclusão e exclusão. Quando necessário, um pesquisador com experiência na área foi consultado para se chegar a um consenso sobre a aceitação dos estudos identificados.

3.1.9.3 Validade Externa

Outra possível ameaça é que alguns estudos não puderam ser encontrados porque não contém palavras-chave definidas na *String* de pesquisa. Para minimizar esta ameaça, o livro "*Handbook on Agent-Oriented Design Processes*" (COSSENTINO et al., 2014c) foi usado como fonte de pesquisa e alguns artigos clássicos foram selecionados manualmente por um especialista na área. Para complementar a pesquisa foi realizada uma busca manual nas metodologias encontradas com o objetivo de garantir a utilização de estudos com a versão mais recente.

3.1.9.4 Validade de Cobertura

Em relação aos possíveis artigos que não foram capturados pelo nossa *String*, pretendemos, como trabalho futuro, aplicar a técnica de bola de neve (*snowballing*) buscando encontrar artigos mais relevantes. Outra questão é que a técnica de bola de neve (*snowballing*) pode nos permitir encontrar mais artigos sobre as metodologias analisadas, uma vez que nesta análise focamos apenas no último artigo de cada metodologia e esta prática pode não garantir uma cobertura completa da metodologia.

3.1.9.5 Validade de Conclusão

Apesar de seguir um protocolo sistemático, as revisões sistemáticas estão sujeitas ao erro humano, principalmente na extração de dados de artigos. Para mitigar essa ameaça, a extração de dados foi realizada por dois pesquisadores independentes seguindo a estratégia definida na subseção 3.1.7 e, em caso de divergências, um especialista da área foi consultado.

3.2 Outros Trabalhos Posteriormente Encontrados

Durante a condução deste estudo, localizamos outros trabalhos relacionados à especificação de requisitos para sistemas multiagentes. Na Revisão Sistemática da Literatura, apresentada no Capítulo 3, estávamos interessados em processos/metodologias que apresentavam a subárea de especificação de requisitos com o objetivo de encontrar lacunas nesses estudos em relação ao suporte ao modelo BDI. No entanto, para tentar garantir uma maior cobertura de estudos que apresentam especificação de requisitos para SMA buscamos novamente estudos que apresentavam a palavra chave *"Requirements Specifica-*

tion" (Especificação de Requisitos) juntamente com a palavra chave *"Multi-agent"* e suas variações, semelhante à *"String"* utilizada na RSL.

O primeiro estudo relevante, foi proposto por Heinze, Papasimeon e Goss (2000), em que foi desenvolvida uma metodologia que estende a UML (UML..., 2021) para modelagem de casos de uso. No entanto, essa representação não suporta o modelo BDI. Diferente da notação proposta pela MASRML (GUEDES et al., 2020) que procura representar características deste modelo.

Já o estudo apresentado por Slhoub, Carvalho e Bond (2017), propõe a extensão do modelo (*template*) do padrão IEEE 830 (IEEE..., 1998) para especificar sistemas multiagentes. No entanto, o padrão utilizado já está desatualizado e foi substituído pelo padrão ISO/IEC/IEEE 29148:2018 (ISO/IEC/IEEE..., 2018). Além disso, a proposta de Slhoub, Carvalho e Bond (2017) não permite a especificação de características do modelo BDI.

A seguir detalharemos cada um desses estudos. Eles apresentam estudos sobre especificação de requisitos para Sistemas Multiagentes semelhante ou que serviram de inspiração para o desenvolvimento de nossa especificação de requisitos.

- ***Recommended Practices for the Specification of Multi-Agent Systems Requirements*** (SLHOUB; CARVALHO; BOND, 2017):

No estudo apresentado por Slhoub, Carvalho e Bond (2017), é proposta uma adaptação do padrão IEEE... (1998) para especificação de requisitos para sistemas multiagentes. Tendo como motivação a afirmação de Wooldridge e Jennings (1995) o qual afirma que as abordagens orientadas a agente podem significar um aprimoramento de nossa capacidade de modelar, projetar e construir sistemas complexos e *softwares* distribuídos.

Segundo o autor, ao contrário do software convencional, a fase de desenvolvimento de um SMA muitas vezes precisa incluir detalhes complexos de agentes de software, seus papéis, suas interações dinâmicas, seus padrões de cooperação e comportamentos combinados com outros detalhes de seus ambientes físicos/virtuais ou operacionais, e o domínio do aplicativo alvo.

Além disso, como a aplicação de diferentes tipos de padrões estava sendo utilizada para aumentar a qualidade do desenvolvimento de software convencional, os autores Slhoub, Carvalho e Bond (2017) decidiram utilizar o IEEE... (1998) para melhorar a formalização do processo de análise de requisitos em SMA e para que a indústria percebece os benefício deste tipo de sistema.

Dessa forma, os autores propuseram uma série de extensões ao padrão IEEE... (1998). Esta adaptação segue uma abordagem orientada a modelos, com isso, os autores analisaram os principais modelos de dados do processo de especificação de requisitos em diferentes metodologias, tais como PASSI (COSSENTINO; SEIDITA, 2014), ADELFE

(BONJEAN et al., 2014), MaSE (WOOD; DELOACH, 2001), Gaia (COSSENTINO et al., 2014b) e Tropos (MYLOPOULOS; CASTRO; KOLP, 2013).

Após análise destas metodologias, os autores selecionaram cinco modelos de dados necessários para especificar os requisitos de SMA, a saber:

- Modelo de Domínio;
- Modelo de Papel;
- Modelo de Agente;
- Modelo de Interação; e
- Modelo de Conhecimento.

O Modelo de Domínio se preocupa com a análise do mundo real específico em que um SMA se destina a trabalhar e interagir. Este modelo especifica três tipos de informações: I) o ambiente operacional, que inclui as descrições das características gerais do ambiente operacional do SMA. Ele fornece a infraestrutura necessária na qual os agentes são implantados e também gerencia esses agentes durante a execução de suas tarefas. Além disso, é responsável por fornecer as estruturas da linguagem de comunicação do agente; II) o ambiente físico/virtual, que inclui descrições do mundo que os agentes de software percebem e com o qual interagem; e III) o domínio do aplicativo, que inclui informações sobre o campo em que o sistema opera.

O Modelo de Papel, está preocupado com as descrições dos requisitos do SMA, em termos de tarefas do agente ou serviços. Os agentes são os atores principais responsáveis por executar tarefas e alcançar objetivos delegados. As descrições podem incluir informações como as responsabilidades do papel, pré-requisitos e restrições.

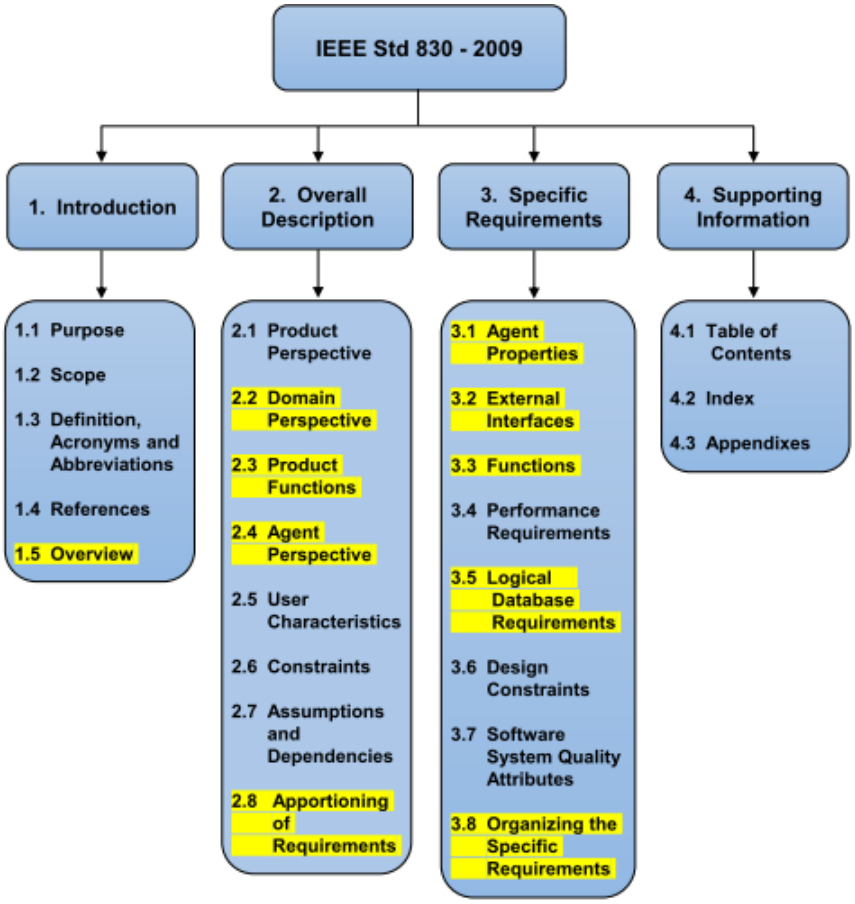
Já o Modelo de Agente, se preocupa em identificar e classificar os tipos de agentes que são iniciado mais tarde no tempo de execução do sistema, e está diretamente ligado ao modelo de papel. Este modelo também deve incluir descrições de características para cada tipo de agente que pode incluir descrições das características essenciais dos agentes, tais como autonomia, reatividade, pró-atividade e sociabilidade

O Modelo de Interação descreve o comportamento dos agentes em termos de cooperação, colaboração, negociando com usuários, recursos do sistema e outros agentes. Também deve especificar os protocolos de mensagens dos agentes e padrões de interação. Isso envolve o uso de ontologias ou meios conceituais para descrever o conteúdo da mensagem.

Já o Modelo de Conhecimento, descreve um repositório de conhecimento que os agentes podem usar para fornecer explicações, informações de recuperação ou otimizar seus desempenhos. Por exemplo, alguns tipos de agente, como agentes reflexivos e os agentes de aprendizagem, requerem algumas regras particulares para executar ações, então

tais regras precisam ser especificadas e armazenadas no sistema. Este modelo também deve descrever brevemente a arquitetura do agente.

Figura 3 – Estrutura geral do padrão IEEE Std 830 adaptado.



Fonte: Slhoub, Carvalho e Bond (2017).

Dessa forma, os autores adicionaram novas seções no IEEE... (1998) para que os cinco modelos apresentados anteriormente sejam descritos, conforme destaque na Figura 3. Sendo que a subseção "Perspectiva de Domínio" foi estendida para descrever o "Modelo de Domínio". Já a subseção "Funções do Produto" do próprio padrão foi adaptada para que o "Modelo de Papel" seja descrito. Para a descrição do "Modelo de Agente" foi estendida a subseção de "Perspectiva de Agente". Já a subseção "Interfaces Externas" existente no próprio padrão foi adaptada para descrição do "Modelo de Interação". Por fim, a subseção "Requisitos Lógicos de Banco de Dados" foi adaptada para descrição do "Modelo de Conhecimento".

No entanto, os modelos e novas seções criadas pelos autores não permitem a identificação de características necessárias para o modelo BDI. Também observamos que o estudo não apresenta de que forma os artefatos criados são verificados e validados. Além disso, o padrão utilizado pelo autor já está desatualizado sendo substituído pelo padrão ISO/IEC/IEEE 29148:2018 que está atualmente em uso.

- ***Specifying Agent Behaviour with Use Cases*** (HEINZE; PAPASIMEON; GOSS, 2000):

Os autores Heinze, Papasimeon e Goss (2000), propõem uma metodologia baseada nas técnicas de modelagem de caso de uso fornecidas pela UML (UML. . . , 2021). Segundo os autores, a UML foi estendida para os agentes na Agent UML (AUML) (BAUER, 1999), no entanto, ela concentrava-se na documentação das interações dos agentes.

O uso da UML é justificado pelos autores porque permite uma visão da funcionalidade do sistema na perspectiva do usuário. Além disso, os requisitos podem ser especificados em termos não técnicos que um usuário pode entender sem precisar ser um engenheiro de software.

Como os agentes são diferenciados da visão dos atores no caso de uso tradicional por serem internos aos sistema, os autores propõem o uso do mecanismo de extensão de estereotipagem da UML, dessa forma, duas extensões para especificação de requisitos são propostas.

Os agentes são especificados como «*Agent*» e devem estar confinados dentro dos limites do sistema. Já os casos de uso acessados por agentes são especificados como «*Use Case Agent*». No entanto, conforme relatado pelos autores a notação diagramática de um «*Use Case Agent*» é a mesma para um caso de uso da UML padrão, deste modo, no estudo não ficou claro de que forma os casos de uso de agentes serão especificados.

Os autores também propõem uma estrutura para a documentação dos casos de uso. Esta estrutura é compreendida pelas seguintes informações:

- Nome do caso de uso: descreve o nome do caso de uso;
- Texto descritivo: descreve o comportamento do agente no contexto do ambiente do agente (interação com outros objetos) e interação com outros agentes e atores;
- Agentes: descreve uma lista dos agentes envolvidos no caso de uso que está sendo detalhado;
- Atores: descreve uma lista dos atores envolvidos no caso de caso de uso que está sendo detalhado;
- Caso de uso associado: Uma lista de outros casos de uso associados a este caso de uso;
- Ambiente: descreve o ambiente e os objetos no ambiente com os quais o agente interage neste caso de uso. Um diagrama de classe UML geralmente é útil nesta seção para mostrar como o agente se ajusta ao ambiente.
- Pré-condições: descreve uma lista de todas as condições que devem ser consideradas verdadeiras antes do agente pode iniciar o caso de uso que está sendo detalhado;

- Pós-condições: descreve uma lista de todas as condições que devem ser consideradas verdadeiras após o agente concluir um comportamento definido pelo caso de uso que está sendo descrito;

- Fluxo de eventos: descreve uma lista numerada de atividades ou eventos que o agente inicia, que define o comportamento do agente neste caso de uso. Esta seção também deve incluir um diagrama de atividades UML que representa o fluxo de eventos. O uso de raias mostra interações nos eventos de fluxo com outros agentes. O uso de sequência acuml e diagramas de colaboração também podem ser usados para visualizar a interação do agente com as entidades ou objetos no ambiente de forma semelhante à sugerida por Odell, Parunak e Bauer (2000).

- Fluxos alternativos: descreve uma lista numerada de atividades ou eventos que os agentes atuam em circunstâncias anômalas ou excepcionais. Um fluxo alternativo descreve o comportamento do agente neste caso de uso não tratado pelo fluxo principal de eventos. É possível ter mais de um fluxo alternativo em um caso de uso. Esta seção também deve incluir diagramas de atividades, sequência e colaboração.

Os diagramas de sequência e atividades são utilizados conforme proposto na AUML. Ou seja, os autores não propõem modificações na estrutura da AUML e também não apresentam nenhuma modificação para a diagramação dos casos de uso. Além disso, observamos que a especificação de requisitos proposta pelos autores não apresenta a representação de características do modelo BDI.

4 ADAPTAÇÃO DA PBR PARA INSPECIONAR DOCUMENTOS DE ESPECIFICAÇÃO DE REQUISITOS

A verificação de requisitos é crucial porque os erros identificados nos requisitos em fases posteriores do desenvolvimento de software tem um custo maior para corrigir. Qualquer mudança em requisitos na fase posterior significa que você precisará fazer alterações no projeto, na arquitetura ou na implementação (BILAL et al., 2016). Além disso, segundo Anu et al. (2019) existe uma forte necessidade de escrever corretamente os requisitos, pois o resultado de requisitos incorretos será a falha sistemática na construção de produtos de software.

Na subárea de especificação de requisitos, os inspetores podem detectar requisitos inconsistentes para o sistema antes que eles sejam passados para os projetistas ou programadores, o que exigiria retrabalho (EBAD, 2017). As inspeções de software provaram ser um meio eficaz de encontrar falhas em diferentes artefatos de software, e se acredita que a aplicação de inspeções de software nas especificações de requisitos oferece um alto retorno sobre o investimento, pois os problemas são detectados precocemente (FOGELSTRÖM; GORSCHEK, 2007).

Garantir que o documento de especificação de requisitos de software tenha a qualidade necessária é fundamental para o sucesso de qualquer projeto de desenvolvimento de software, uma vez que suas informações são utilizadas em todas as etapas do projeto (SAAVEDRA; BALLEJOS; ALE, 2013).

O estudo apresentado por Ebad (2017), avaliou as evidências sobre a eficácia da técnica de leitura para inspecionar os artefatos produzidos durante as fases de requisitos, projeto e codificação. Na fase de requisitos, segundo o autor, a técnica *ad hoc* é a forma mais simples das técnicas de leitura, mas nela os inspetores não seguem diretrizes ou orientações. Nesse mesmo estudo, o autor destaca que a PBR é uma das técnicas mais eficazes para inspeção de requisitos.

Já a inspeção proposta por Fagan (1976), inclui a ideia de usar listas de verificação (Checklists) para detecção de defeitos. A Leitura Baseada em Lista de Verificação (CBR), é mais sistemática que a *ad hoc* e pode ser eficaz em detectar defeitos específicos, como consistência nos artefatos de requisitos. Já os autores Porter e Votta (1994), apresentaram uma técnica baseada em cenários, que fornece aos inspetores instruções mais específicas que a lista de verificação. Por meio dessa técnica, os defeitos são classificados e um conjunto de questões é desenvolvido para cada classe de defeito.

O estudo de Thelin et al. (2004) introduziu a Leitura Baseada no Uso (UBR), no qual os inspetores usam uma lista pré-definida de casos de uso priorizados, essa técnica foi aplicada na inspeção de projeto. Já o UBR-ir (THELIN; RUNESON; WOHLIN, 2003), uma variação do UBR, permite que os inspetores classifiquem individualmente os casos de uso no início do processo de inspeção de acordo com a experiência e compreensão dos inspetores. Já o estudo de Dunsmore, Roper e Wood (2003), apresentou a leitura de caso

de uso (UCR) para inspecionar sistemas orientados a objetos levando em consideração a interação dinâmica com objetos colaborativos.

O que todas essas técnicas tem em comum é que foram utilizadas em sistemas tradicionais, principalmente os orientados a objetos. Todavia, segundo Padmanaban et al. (2016), testes de software convencionais orientados a objetos não podem ser aplicados aos sistemas orientados a agentes devido às propriedades do agente, como autonomia, proatividade, capacidade social, reatividade e mobilidade.

Dessa forma, considerando as evidências apontadas por Ebad (2017) que a PBR é uma das técnicas mais eficazes para inspecionar requisitos, decidimos adaptar essa técnica para a inspeção de documentos de especificação de requisitos. Nesse sentido, Shull, Rus e Basili (2000) nos ensinam que dependendo do ambiente onde a PBR é aplicada, se pode encontrar um conjunto diferente de perspectivas mais aplicáveis. Esses mesmos autores, afirmam que este conjunto de perspectivas mais apropriado é uma maneira de adaptar a PBR a um ambiente específico.

Uma das perspectivas propostas na PBR tradicional, é a do Operador. Essa perspectiva tem como propósito verificar as funcionalidades essenciais do sistema na visão do usuário e possíveis inconformidades identificadas nelas. Não tem o propósito de verificar requisitos técnicos inerentes aos projetistas e testadores.

O PBR-Operador está interessado em saber se os requisitos descrevem adequadamente as funcionalidades do sistema acessadas por ele. O leitor deve ler a especificação de requisitos e a partir dessas descrições e verificar se é possível produzir diagramas de casos de uso. Esses diagramas de casos de uso tem o objetivo de identificar se as funcionalidades contidas neles representam o que usuário vai acessar no sistema.

Na PBR a perspectiva do Operador, busca inspecionar a especificação de requisitos com base na perspectiva do usuário e somente o que ele acessa no sistema. No entanto, as funcionalidades executadas por agentes não estarão cobertas por essa perspectiva, uma vez que os agentes executam ações de forma autônoma sem a necessidade de intervenção humana. Além disso, em diversos sistemas os agentes agem sem que sejam percebidos pelos usuários normais, que muitas vezes nem sabem que existem agentes operando no sistema.

Portanto, como os agentes não são humanos, inspecionar as funcionalidades dos agentes a partir da visão do Operador violaria as diretrizes da PBR. Pois ela parte da premissa que as perspectivas são na visão de partes envolvidas. Para resolver este problema seguindo as diretrizes da PBR, propomos que uma parte envolvida assuma a perspectiva do agente para inspecionar o documento de especificação de requisitos de um SMA. Para isso, usamos como metáfora a perspectiva do simulador de agente, que denominamos de PBR-Simulador de Agente. Dessa forma uma pessoa assumirá a perspectiva do agente com o objetivo de simular as ações de um agente dentro do sistema.

Com isso, nesse trabalho, estamos propondo a perspectiva de Simulador de Agente

para inspeção do Documento de Especificação de Requisitos de Software (DERS) considerando características específicas de sistemas multiagentes. A perspectiva do PBR-Simulador de Agente, procura determinar se os requisitos descrevem adequadamente as funcionalidades executadas por agentes. Essa perspectiva inspeciona o *Internal Use Case* ou Caso de Uso Interno (IUC), observando se os cenários principais e cenários alternativos estão descritos, quando houver se as pré-condições estão definidas.

Além disso, com base na perspectiva de Simulador de Agente é possível inspecionar se os papéis de agentes estão identificados e detalhados, se as crenças iniciais estão descritas, se as percepções estão associadas aos objetivos e verificar a corretude dos estereótipos ou associações dos casos de uso internos.

A diferença da nossa nova perspectiva para a perspectiva do Operador da PBR tradicional é que, o Operador está interessado em inspecionar as funcionalidades acessadas e percebidas por usuários normais, ou seja, humanos. Já a perspectiva do Simulador de Agente, tem como objetivo inspecionar as funcionalidades internas do sistema que são acessadas por agentes autônomos.

Para inspeção seguindo a perspectiva de PBR-Simulador de agente, foi definida uma sequência de passos, conforme descreveremos a seguir na seção 4.1. Esses passos, serão seguidos pelo inspetor que simulará as ações dos agentes para localizar erros ou falhas no documento de especificação de requisitos.

Para registrar as falhas ou erros encontrados no DERS, o inspetor deve preencher o Formulário de Inconformidades, conforme apresentado na Figura 4. Este documento conterá, entre outras informações, a localização da falha no DERS e o tipo de falha. O Formulário de Inconformidades foi inspirado no formulário de desconformidades proposto por Pagliuso, Tambascia e Villas-Boas (2002).

O Formulário de Inconformidades apresentado na Figura 4, deve ser rigorosamente preenchido. O inspetor deverá descrever de forma clara e completa as inconformidades encontradas, identificando a página e linha do documento onde a falha foi localizada, qual o tipo de inconformidade encontrada segundo a taxonomia proposta e o número da questão da Lista de Verificação que originou a inconformidade.

Para utilização da técnica de inspeção proposta nesse estudo, foi estabelecida como referência uma taxonomia de falhas a ser explorada pela mesma. A taxonomia utilizada pela PBR-Simulador de Agente é mostrada na Tabela 9.

As demais perspectivas propostas pela PBR tradicional poderão ser utilizadas em conjunto com a perspectiva proposta nesse estudo para inspeção de especificação de requisitos para sistemas multiagentes. No entanto, não aplicamos as demais perspectivas nos quase-experimentos que apresentaremos no Capítulo 6, pois nosso foco é na engenharia de requisitos e a fase de projeto não faz parte do escopo deste estudo.

Segundo Shull, Rus e Basili (2000), normalmente um processo de inspeção tem várias fases: planejamento, visão geral, detecção de defeitos, coleta de defeitos, correção

Figura 4 – Formulário de Inconformidades encontradas.

FORMULÁRIO DE INCONFORMIDADES	
Doc. de Especificação de Requisitos (código, nome, versão):	
Nome Arquivo:	
Elaborador (papel, nome, e-mail):	
Data de Revisão (DD/MM/AAAA): _____/_____/_____	Tempo de Revisão (HH:MM): _____:_____

INCONFORMIDADES ENCONTRADAS				
Nº Questão	Página	Linha	*Tipo	Descrição
<01.1>	<00>	<00>	<ICM>	
OBSERVAÇÕES GERAIS DOS REVISORES				
1. 2. 3.				
*Sigla da Taxonomia: ICM- Incompleto, ICR-Incorreto, ICS-Inconsistente, NRA-Não rastreável, ONL-Outro não listado.				

Fonte: Adaptado de Pagliuso, Tambascia e Villas-Boas (2002).

Tabela 9 – Taxonomia de falhas utilizada pela PBR-Simulador de Agente.

Tipo	**ID**	**Descrição**
Incompleto	ICM	Informações importantes da funcionalidade ou caso de uso interno omitidas no DERS.
Inconsistente	ICS	Duas frases contidas no DERS se contradizem diretamente ou expressam ações que não podem ser corretas ou não podem ser executadas.
Incorreto	ICR	Denota informação que não é verdadeira para as condições especificadas.
Não rastreável	NRA	As funcionalidades ou casos de uso internos não possuem identificadores únicos que permitam a rastreabilidade e histórico de alterações.

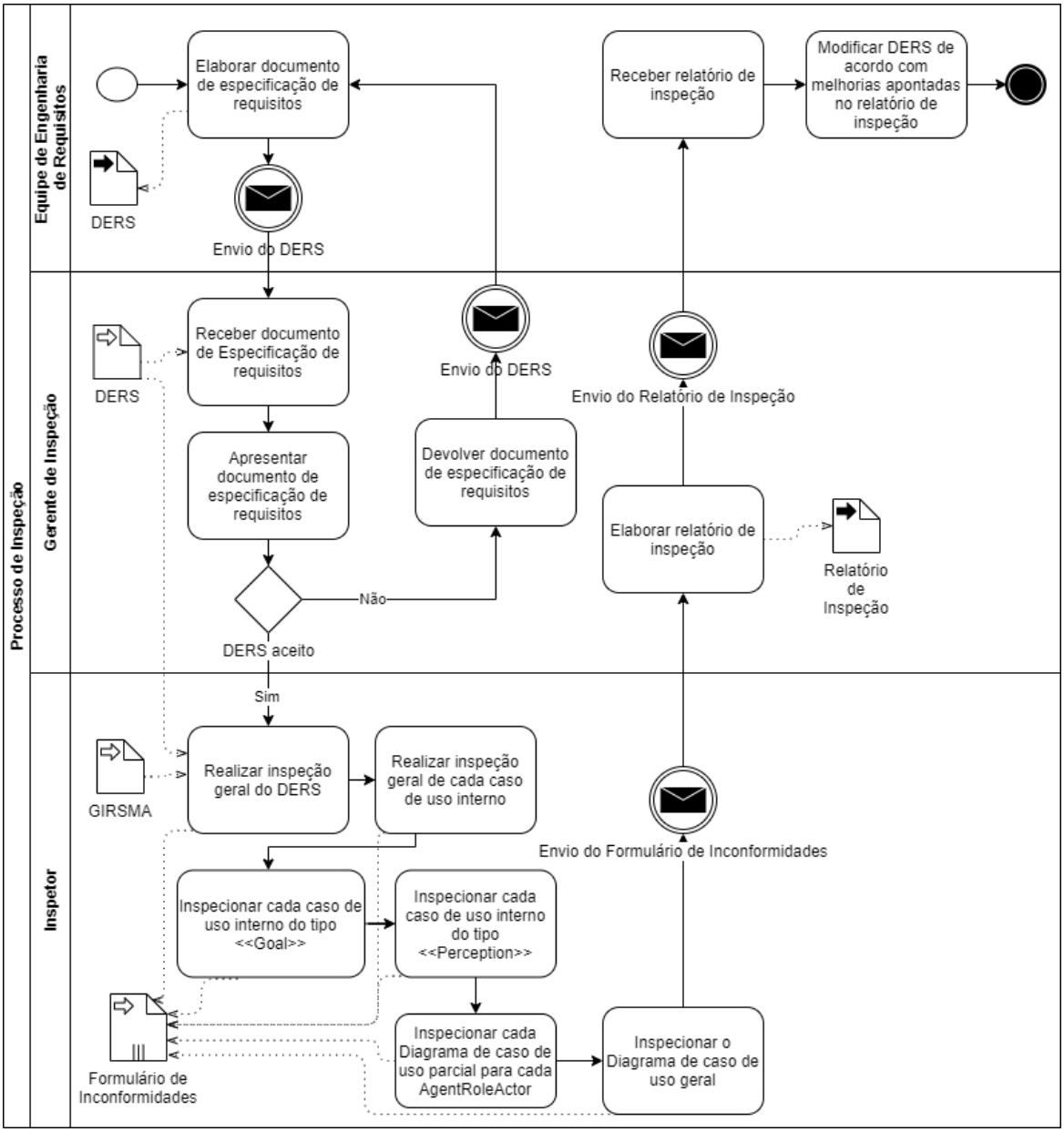
Fonte: Autor.

de defeitos e acompanhamento. Dessa forma, desenvolvemos um Guia de Inspeção de Requisitos para Sistemas Multiagentes (GIRSMA) composto por um processo de inspeção e uma sequência de seis passos para habilitar os inspetores para verificação do documento

de especificação de requisitos, o GIRSMA pode ser consultado no ANEXO C.

O processo compreende um conjunto de atividades que devem ser seguidas para condução da inspeção de requisitos para sistemas multiagentes. O processo de inspeção para SMA pode ser visualizado na Figura 5.

Figura 5 – Processo de inspeção de requisitos para SMA.



Fonte: Autor.

Conforme pode ser observado na Figura 5, o processo inicia com a elaboração do Documento de Especificação de Requisitos (DERS) pela equipe de engenharia de requisitos. Este DERS então é enviado para o responsável pela equipe de inspeção, denominado de gerente de inspeção. Na sequência, o de gerente de inspeção recebe o documento de especificação de requisitos e prepara a reunião de apresentação. Com a equipe reunida o documento de especificação é apresentado e uma análise preliminar é realizada, que tem

como resultado aprovar ou recusar o documento para inspeção considerando a estrutura geral do documento, verificando se as seções estão completas. Caso o documento seja reprovado, ele é devolvido para a equipe de engenharia de requisitos. Se o documento for aprovado ele é distribuído para os inspetores iniciarem a inspeção.

Para cada uma das atividades executadas pelo inspetor desenvolvemos listas de verificações para a perspectiva da PBR-Simulador de Agente. As diretrizes de inspeção permitem aos inspetores identificarem claramente os pontos de inspeção. Elas permitem que os inspetores examinem sistematicamente cada seção de uma especificação de requisitos. Além disso, segundo Saito et al. (2013), a diretriz de inspeção permite a inspeção do documento de especificação de requisitos por pessoas com pouco conhecimento sobre o sistema e domínio. Nesta etapa de nossa proposta os requisitos serão verificados considerando os casos de uso internos.

É difícil inspecionar o documento de especificação de requisitos sem impacto de fatores humanos, além disso, é comum equipes de inspeção terem pouco conhecimento sobre o histórico de projeto e domínio da aplicação (SAITO et al., 2013). Dessa forma, desenvolvemos uma sequência de seis passos para habilitar os inspetores para verificação do documento de especificação de requisitos. Cada passo contém uma lista de verificação que deve ser seguida durante a condução da inspeção. Detalharemos a sequência de seis passos na seção 4.1.

Conforme pode ser observado na Figura 5, quando a etapa de inspeção é finalizada, o gerente de inspeção reúne toda a equipe e elabora o relatório de inspeção. Nesta etapa os pontos de melhorias identificados pelos inspetores são compilados para compor o relatório de inspeção. Este documento descreve as principais descobertas e percepções sobre as causas de inconformidades no documento de especificação de requisitos. A próxima etapa a ser executada é o Parecer, que tem como objetivo enviar o relatório de inspeção para a equipe de engenharia de requisitos.

A engenharia de requisitos é uma atividade criativa em que os engenheiros de software estão tentando construir uma solução para um novo problema, ou seja, uma nova necessidade dos usuários, dessa forma erros são comuns durante essa atividade (ANU et al., 2019). Consequentemente, é necessário utilizar técnicas adequadas para inspeção de documentos de requisitos.

Segundo Boyarchuk et al. (2020), as ferramentas, técnicas e modelos conhecidos não resolvem o problema para verificar a qualidade da especificação de requisitos com base na ISO/IEC/IEEE 29148:2018 (ISO/IEC/IEEE..., 2018). Da mesma forma, até onde vai nosso conhecimento da literatura pesquisada, nenhum outro trabalho apresenta adaptação da Leitura Baseada em Perspectiva (PBR) para inspeção da especificação de requisitos de sistemas multiagentes.

Nesse sentido, a proposta apresentada nessa seção constitui uma nova abordagem, tendo em vista que, propomos uma nova perspectiva para inspeção da especificação de

requisitos para sistemas multiagentes. No entanto, nossa adaptação da técnica PBR é proposta para inspeção do documento de especificação de requisitos produzido pela especificação de requisitos adaptada e apresentada a seguir no Capítulo 5. Seu uso é exclusivo para um processo específico de Engenharia de Requisitos para Sistemas Multiagentes que está sendo desenvolvido em paralelo a este estudo.

4.1 Passos da inspeção para PBR-Simulador de Agente

O inspetor assumirá a perspectiva do agente, dessa forma estará preocupado em inspecionar os casos de uso interno dos agentes. Inspecionará as funcionalidades executadas por agentes considerando sua autonomia e execução sem intervenção de humanos. Para isso, utilizará as listas de verificações a seguir, como guia para identificar falhas ou erros no documento de especificação de requisitos. A cada erro ou falha encontrada o inspetor o registrará no Formulário de Inconformidades, conforme modelo apresentado na Figura 4.

A Lista de Verificação da perspectiva do PBR-Simulador de Agente é composta por uma sequência de seis passos:

- Primeiro Passo - corresponde a inspeção geral do documento de especificação de requisitos;
- Segundo Passo - corresponde a inspeção geral de cada caso de uso interno;
- Terceiro Passo - nesse passo serão inspecionados os caso de uso interno do tipo «*Goal*»;
- Quarto Passo - serão inspecionados os caso de uso interno do tipo «*Perception*»;
- Quinto Passo - serão inspecionados cada diagrama de caso de uso parcial para cada *AgentRoleActor*; e
- Sexto Passo - corresponde à inspeção do diagrama de caso de uso geral.

A seguir descreveremos as listas de verificações para cada passo de nossa técnica de inspeção.

- **Primeiro passo:**

Nesta etapa, o documento de especificação de requisitos é inspecionado de forma geral. O inspetor buscará falhas do tipo completude, corretude e consistência. Algumas delas estão relacionadas à definição de papéis de agentes, as funcionalidades associadas aos agentes, padrão de comunicação utilizado e sobre as crenças identificadas. Apresentamos na Tabela 10, a Lista de Verificação para a inspeção geral do documento de especificação de requisitos.

Tabela 10 – Lista de verificação do primeiro passo.

Nº	Questão	Regra	Opção	
1.1	Os papéis de agente para o sistema foram definidos?	Completude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
1.2	As descrições dos papéis de agente estão claras?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
1.3	Foram atribuídas funcionalidades aos papéis de agente?	Consistência	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
1.4	As funcionalidades atribuídas ao papel de agente estão claras?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
1.5	Existe alguma forma de comunicação entre os agentes e usuários externos?	Completude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
1.6	Foi estabelecido qual o tipo de informação a ser transmitida entre os agentes e usuários externos?	Consistência	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
1.7	Existe alguma forma de comunicação entre os agentes?	Completude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
1.8	Foi definido o padrão de comunicação dos agentes?	Completude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
1.9	Foi estabelecido qual o tipo de informação a ser transmitida entre os agentes?	Consistência	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
1.10	A crença identificada está atribuída a um objetivo?	Completude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
1.11	A descrição da crença está clara?	Consistência	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
1.12	A descrição da crença está duplicada?	Consistência	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
1.13	A descrição da crença está conflitante com outra crença?	Consistência	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.

Fonte: Autor.

- **Segundo passo:**

Nesta etapa, o inspetor deve analisar cada IUC buscando falhas do tipo corretude, não ambiguidade, rastreabilidade, completude e consistência. Algumas dessas falhas estão relacionadas à estrutura geral de cada IUC, falta de cenário principal, ambiguidades ou corretude dos textos dos casos de uso.

Apresentamos na Tabela 11, a Lista de Verificação para a inspeção de cada IUC descrito na Seção 3.11 do documento de especificação de requisitos, para verificar a estrutura da especificação de requisitos de software para SMA consulte o Capítulo 5.

Tabela 11 – Lista de verificação geral para o caso de uso interno do segundo passo.

Nº	**Questão**	**Regra**	**Opção**	
2.1	O texto do caso de uso interno apresenta erros de português?	Corretude	Sim ( ) *Informe no Formulário de Inconformidades.	Não ( )
2.2	O texto do caso de uso interno está escrito de forma ambígua?	Ambiguidade	Sim ( ) *Informe no Formulário de Inconformidades.	Não ( )
2.3	O caso de uso interno possui um código para identificação?	Rastreabilidade	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
2.4	O nome do caso de uso interno inicia com verbo de ação no infinitivo?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
2.5	Os papéis de agente (*AgentRoleActor*) associados ao caso de uso interno estão identificados?	Completude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
2.6	O papel do agente que interage com o caso de uso interno está descrito na seção adequada?	Completude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
2.7	O caso de uso interno apresenta resumo que descreve de forma clara o propósito do caso de uso interno?	Completude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
2.8	Se o caso de uso interno possuir crenças iniciais, essas crenças entram em conflito com outras crenças?	Consistência	Sim ( ) *Informe no Formulário de Inconformidades.	Não ( )
2.9	Se a Pré-condição exige a execução de outro caso de uso interno, este está descrito separadamente?	Consistência	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
2.10	O caso de uso interno tem um cenário principal?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
2.11	Se um cenário principal exige a execução de outro caso de uso interno, este está descrito separadamente?	Consistência	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
2.12	Se um cenário alternativo exige a execução de outro caso de uso interno, este está descrito separadamente?	Consistência	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.

Fonte: Autor.

- **Terceiro passo:**

Nesta etapa, o inspetor deve analisar cada um dos casos de uso internos do tipo «*Goal*», buscando falhas do tipo corretude, consistência e completude. Algumas delas estão relacionadas à corretude do estereótipo, clareza na descrição das percepções que tornam o objetivo uma intenção ou existência de percepções associadas ao objetivo. Apresentamos na Tabela 12, a Lista de Verificação para a inspeção de cada IUC do tipo «*Goal*».

Tabela 12 – Lista de verificação para caso de uso interno do tipo «*Goal*» do terceiro passo.

Nº	**Questão**	**Regra**	**Opção**	
3.1	Casos de uso internos que representam desejo (*Goal* - Objetivo) dos papéis de agentes estão identificados com esterótipo do tipo *Goal*?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
3.2	A(s) percepção(ões) que torna(m) o objetivo (*Goal*) uma intenção está(ão) descrita(s) de forma clara?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
3.3	É possível estabelecer critérios de prioridades de desejos (*goals*/objetivos) a serem atingidos/satisfeitos?	Consistência	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
3.4	É necessário um cenário alternativo para o objetivo (*Goal*) se tornar uma intenção?	Completude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
3.5	A(s) crença(s) inicial(ais) esta(ão) descrita(s)?	Completude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
3.6	Existem percepções associadas ao objetivo?	Completude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
3.7	Está claro como o objetivo se torna uma intenção?	Consistência	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
3.8	Existe um cenário em caso de falha na execução de uma intenção?	Consistência	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.

Fonte: Autor.

- **Quarto passo:**

Nesta etapa, o inspetor analisará cada um dos casos de uso internos do tipo «*Perception*», buscando falhas do tipo corretude e completude. Algumas delas estão relacionadas à corretude do estereótipo, identificação e clareza das crenças iniciais e definição de pré-condições. Apresentamos na Tabela 13, a Lista de Verificação para a inspeção de cada IUC do tipo «*Perception*».

Tabela 13 – Lista de verificação para caso de uso interno do tipo «*Perception*» do quarto passo.

Nº	**Questão**	**Regra**	**Opção**	
4.1	O estereótipo do caso de uso interno foi definido como Perception?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
4.2	As Pré-condições estão definidas?	Completude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
4.3	A(s) crença(s) inicial(ais) esta(ão) descrita(s)?	Completude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
4.4	As crenças iniciais estão descritas de forma clara?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.

Fonte: Autor.

- **Quinto passo:**

Nesta etapa o inspetor deve analisar cada diagrama de caso de uso parcial elaborado para cada *AgentRoleActor*. Ele buscará falhas do tipo corretude e consistência. Algumas delas estão relacionadas à corretude dos estereótipos, corretude da associação entre componentes do diagrama e a corretude da direção das associações entre componentes do diagrama. Apresentamos na Tabela 14, a Lista de Verificação para inspeção de cada Diagrama de caso de uso parcial para cada *AgentRoleActor*.

Tabela 14 – Lista de verificação para inspeção de cada Diagrama de caso de uso parcial para cada *AgentRoleActor* do quinto passo.

Nº	**Questão**	**Regra**	**Opção**	
5.1	Os objetivos (*Goals*) associados ao papel de agente estão identificados com o estereótipo do tipo «*Goal*»?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
5.2	Se houver associação entre um caso de uso interno Objetivo (*Goal*) e um caso de uso interno de ação (*Action*), essa associação é do tipo «actionPerceptionInclude»?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
5.3	A direção da associação é do objetivo (*Goal*) para a ação (*Action*)?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
5.4	Se houver associação entre um caso de uso interno Objetivo (*Goal*) e um caso de uso interno de percepção (*Perception*) essa associação é do tipo «actionPerceptionInclude»?	Completude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
5.5	Os casos de uso internos do tipo percepção (*Perception*) estão identificados com o estereótipo «*Perception*»?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
5.6	A direção da associação é do objetivo (*Goal*) para o caso de uso interno do tipo «*Perception*»?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
5.7	O plano (*Plan*) está identificado com o estereótipo do tipo «*Plan*»?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
5.8	O plano (*Plan*) está associado a pelo menos um caso de uso interno do tipo «*Goal*»?	Consistência	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
5.9	A associação do Plano com o caso de uso interno Objetivo (*Goal*) é do tipo «planExtend»?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
5.10	A direção da associação é do plano (*Plan*) para o caso de uso interno do tipo «*Goal*»?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
5.11	A associação do plano (*Plan*) com o caso de uso interno do tipo *Action* é do tipo «actionPerceptionInclude»?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
5.12	A associação do plano com a percepção é do tipo «actionPerceptionInclude»?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
5.13	A direção da associação é do plano (*Plan*) para a ação (*Action*)?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
5.14	A ação (*Action*) está identificada com estereótipo do tipo «*Action*»?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
5.15	A ação (*Action*) está associada a pelo menos um caso de uso interno do tipo «*Plan*» ou «*Goal*»?	Consistência	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.

Fonte: Autor.

- **Sexto passo:**

Por fim, nesta etapa o inspetor deve analisar o diagrama de caso de uso geral, buscando localizar falhas do tipo corretude e completude. Algumas delas estão relacionadas à corretude do estereótipo dos papéis de agentes, papéis de agentes e usuários externos identificados nos locais corretos, e atribuição de pelo menos um objetivo a cada papel de agente. A Lista de Verificação para inspeção do Diagrama de Caso de Uso geral pode ser verificado na Tabela 15.

Tabela 15 – Lista de verificação para inspeção do Diagrama de Caso de Uso geral do sexto passo.

Nº	**Questão**	**Regra**	**Opção**	
6.1	Os atores que representam os usuários externos estão identificados fora da fronteira do sistema (representada por um retângulo)?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
6.2	Os papéis de agentes estão identificados dentro da fronteira do sistema (retângulo)?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
6.3	Os papéis de agentes estão identificados com estereótipos do tipo ReactiveAgentRole, CognitiveAgentRole, AgentRolePS, AgentRoleUAM ou AgentRoleSMI?	Corretude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.
6.4	Existe pelo menos um objetivo associado a cada papel de agente?	Completude	Sim ( )	Não ( ) *Informe no Formulário de Inconformidades.

Fonte: Autor.

As Listas de Verificações apresentadas nos seis passos, não são configuradas de forma totalmente fechada, elas podem ser complementadas com novas questões elaboradas pela equipe de inspeção conforme a experiência em verificações anteriores. Dessa maneira, nossa proposta de inspeção pode ser adaptada para diferentes tipos de sistemas multiagentes que utilizem o modelo BDI.

Nossa adaptação da técnica PBR foi aplicada em dois quase-experimentos para avaliarmos sua eficácia na detecção de falhas em um documento de especificação de requisitos. Os resultados dos quase-experimentos podem ser verificados no Capítulo 6.

Para aplicar a técnica de inspeção proposta nesse capítulo, foi necessário a produção de um documento de especificação de requisitos que suporte as características do modelo BDI. No entanto, nenhum dos estudos retornados em nossa RSL permitem a produção de tal documento. O estudo mais próximo é o apresentado por Slhoub, Carvalho e Bond (2017) em que é proposta uma extensão do padrão IEEE 830 (IEEE..., 1998) para SMAs, no entanto, ele não suporta o modelo BDI e já está desatualizado. Dessa forma, identificamos que o padrão ISO/IEC/IEEE 29148:2018 (ISO/IEC/IEEE..., 2018) é o mais atual para especificação de requisitos. Assim sendo, nós propomos a extensão do modelo (*template*) de tal padrão para que ele permita a produção de um documento de especificação de requisitos que suporte as características do modelo BDI. Nossa proposta de extensão do padrão é apresentada a seguir no Capítulo 5.

5 ESPECIFICAÇÃO DE REQUISITOS PARA SISTEMAS MULTIAGENTES

A tecnologia de agente inteligente é empregada no desenvolvimento de aplicações complexas (ABUSHARK et al., 2016), tais como sistemas de simulação militar, logística, planejamento e tutoria inteligente (MüLLER; FISCHER, 2014; MUNROE et al., 2006; SPROCK; GALLEGOS; VICARI, 2017). O desenvolvimento de tais sistemas requer metodologias apropriadas. Segundo Abushark et al. (2016), um aspecto fundamental de todas essas metodologias é a especificação de requisitos.

Segundo Sommerville (2004), a especificação de requisitos descreve o que o sistema deve fazer e suas restrições de desenvolvimento. Nesse sentido, Zhi (2021) afirma que uma especificação de requisitos é uma descrição detalhada de um sistema de software a ser desenvolvido.

Normalmente é difícil lidar com documentos de especificação de requisitos em sua forma escrita, além disso, devido à complexidade geral desses documentos as partes interessadas do software tendem a não usá-los de forma eficiente (ALSANAD; CHIKH, 2014). Para resolver esse problema surgiram diversas abordagens/padrões para padronização da especificação de requisitos, dentre elas está o padrão ISO/IEC/IEEE 29148:2018 (ISO/IEC/IEEE..., 2018).

Segundo Boyarchuk et al. (2020), durante a formação e formulação dos requisitos, é importante cumprir os padrões que regem o processo de desenvolvimento de software e o principal padrão básico para especificar os requisitos de software é o ISO/IEC/IEEE 29148:2018.

Em um estudo recente desenvolvido por Franch et al. (2021), foi realizada uma pesquisa com 90 especialistas em engenharia de requisitos na indústria, com objetivo de analisar o uso de padrões na prática. Nesse estudo, foi constatado que diversos padrões são conhecidos e utilizados. Entre esses padrões os mais citados, respectivamente, foram o IREB *Glossary* (GLINZ, 2013) com 63 dos entrevistados, o IEEE 830 (IEEE..., 1998) com 50 dos entrevistados e o ISO/IEC/IEEE 29148:2018 (ISO/IEC/IEEE..., 2018) com 49 dos entrevistados.

A frequência de uso do IREB *Glossary* em todos os projetos ou na maioria dos projetos corresponde a 40% dos entrevistados. Já o uso em alguns projetos corresponde a pouco mais de 30%. A frequência de uso do IEEE 830 em todos os projetos ou na maioria dos projetos corresponde a 30% dos entrevistados. Já o uso em alguns projetos corresponde a quase 50% dos entrevistados. E a frequência de uso do ISO/IEC/IEEE 29148:2018 em todos os projetos ou na maioria dos projetos corresponde a quase 30% dos entrevistados. Já o uso em alguns projetos corresponde a pouco mais de 40% dos entrevistados.

Ainda, no estudo apresentado por Franch et al. (2021), foi investigado os impedimentos pelos quais os padrões não são usados. A maioria dos entrevistados (43,2%) relataram a falta de conhecimento sobre os padrões como o principal impedimento de

sua adoção. Entre eles estão: I) Conhecimento/habilidades insuficientes; II) Capacidade da empresa; III) Falta de consciência; IV) Falta de especialistas no domínio do aplicativo/projeto; e V) Contexto da empresa, adoção industrial limitada.

O IEEE desenvolve seus padrões por meio de um processo de desenvolvimento aprovado pela *American National Standards Institute* (IEEE..., 1998). Inicialmente foi lançado o padrão IEEE 830 (IEEE..., 1998), que posteriormente foi substituído pelo padrão ISO/IEC/IEEE 29148:2011 (ISO/IEC/IEEE..., 2011), o qual foi cancelado e substituído em 2018, originando o padrão ISO/IEC/IEEE 29148:2018 (ISO/IEC/IEEE..., 2018).

O padrão ISO/IEC/IEEE 29148:2018, fornece diretrizes para os processos e produtos na engenharia de requisitos ao longo do ciclo de vida dos sistemas e softwares. Esse padrão pode ser adotado independentemente da metodologia usada, bem como tamanho ou complexidade do sistema (ISO/IEC/IEEE..., 2018). No entanto, ele não suporta características específicas de sistemas multiagentes, principalmente no que diz respeito ao modelo BDI. Dessa forma, identificamos a necessidade de adaptar esse padrão para representar requisitos específicos de sistemas multiagentes com objetivo de validar nossa proposta de adaptação da técnica PBR apresentada no Capítulo 4.

Segundo DeLoach e Garcia-Ojeda (2014), os sistemas multiagentes não foram amplamente adotados na indústria por razões que incluem falta de métodos e ferramentas para apoiar o desenvolvimento multiagente. Dessa forma, ao adaptar um padrão reconhecido internacionalmente para suportar características específicas de sistemas multiagentes temos a intenção de promover os benefícios da AOSE na indústria. Além disso, como nossa proposta de extensão foi desenvolvida com base em um padrão já utilizado na indústria e academia isso poderá torná-lo mais aceito e compreendido.

No entanto, o modelo (*template*) do padrão ISO/IEC/IEEE 29148:2018 tradicional não permite identificar características específicas para sistemas multiagentes, conforme apresentado na seção 2.8. Dessa forma, nesse estudo estamos propondo uma extensão do modelo (*template*) de tal padrão para suportar esse tipo de sistema com ênfase da representação de requisitos necessários para o modelo BDI. Considerando que, segundo Boufedji et al. (2018) sempre que o agente possui uma arquitetura BDI, todos os recursos de Crenças, Objetivos e Planos são obrigatórios. A seguir destacamos nossas adaptações no modelo (*template*) do padrão ISO/IEC/IEEE 29148:2018.

5.1 Estendendo o padrão ISO/IEC/IEEE 29148:2018 para Sistemas Multiagentes

Nesta subseção detalharemos nossas contribuições para que o modelo (*template*) do padrão ISO/IEC/IEEE 29148:2018 suporte características de Sistemas Multiagentes com ênfase no modelo BDI e a representação de requisitos apresentadas pela MASRML. Dessa forma, permitirá a produção de um Documento de Especificação de Requisitos de

Software que será utilizado como base para aplicar a técnica de inspeção proposta nesse estudo.

Primeiramente, adaptamos a seção 1.2 (Escopo) para que fiquem mais claras as informações capturadas por esta seção. Essas informações já são requeridas pelo próprio padrão, apenas às adaptamos para uma estrutura de subseções. Dessa forma, adicionamos três subseções, a saber:

- Subseção 1.2.1 - "Nome do produto de software": nesta subseção se deve descrever o nome do produto de software que está sendo especificado.
- Subseção 1.2.2 - "O que o produto de software fará": se deve descrever o que o produto de software fará, considerando qual a aplicação final e principais funcionalidades.
- Subseção 1.2.3 - "Aplicação do software": corresponde a descrição dos benefícios mais relevantes propostos pelo software, incluindo seus objetivos.

A estrutura geral da nossa adaptação do padrão ISO/IEC/IEEE 29148:2018, pode ser visualizada na Figura 6.

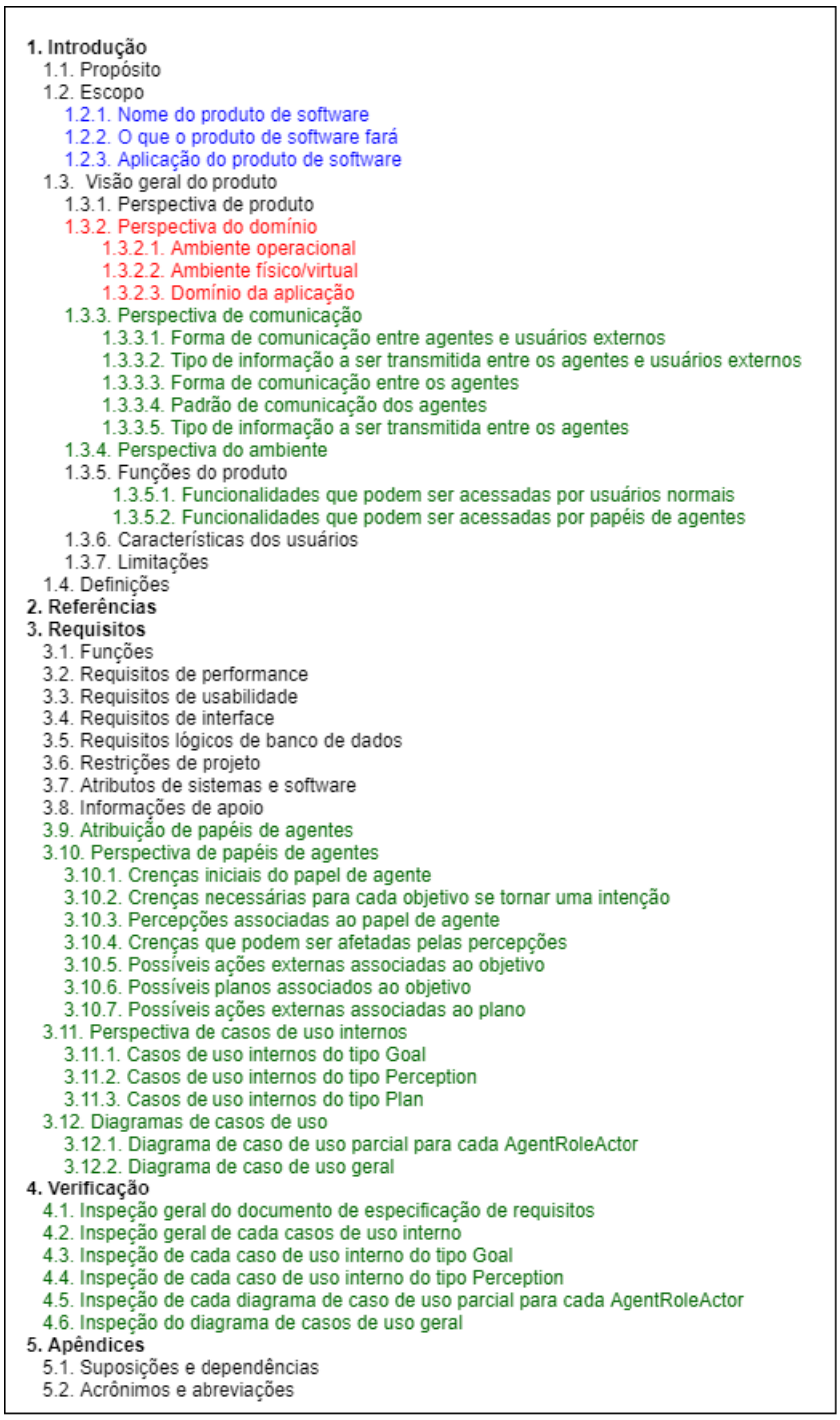
Conforme pode ser observado na Figura 6, definimos uma composição de cores, onde as seções em preto correspondem à estrutura proposta pelo próprio padrão, sem adaptações. As seções em azul correspondem às nossas adaptações no padrão, ou seja, seções já existentes no padrão mas sobre as quais realizamos modificações. As seções em verde correspondem às nossas contribuições, ou seja, novas seções que são necessárias para a especificação de SMAs. E por fim, as seções em vermelho correspondem às contribuições de outros autores em outros estudos sobre especificação de requisitos, as quais consideramos relevantes para adicionar em nossa adaptação.

A próxima adaptação do padrão foi a adição da subseção 1.3.2, que corresponde à "Perspectiva do domínio" proposta originalmente no estudo de Slhoub, Carvalho e Bond (2017). Com isso, adicionamos as seguintes subseções:

- Subseção 1.3.2.1 - "Ambiente operacional": devem ser descritas as principais interações dos usuários com o sistema;
- Subseção 1.3.2.2 - "Ambiente físico/virtual": deve ser detalhado qual o tipo do sistema (como por exemplo *desktop, web ou mobile*), e a descrição das ações dos agentes dentro do sistema; e
- Subseção 1.3.2.3 - "Domínio da aplicação": compreende a descrição de qual o foco do sistema e de que forma os usuários interagem com ele.

No estudo de Slhoub, Carvalho e Bond (2017) é proposta a definição da estrutura de linguagem de comunicação dos agentes na seção denominada "Ambiente operacional".

Figura 6 – Estrutura do padrão ISO/IEC/IEEE 29148:2018 estendido.



Fonte: Autor.

E para especificar a forma de comunicação entre os agentes ele propõe a seção "Interface externa", no entanto, verificamos algumas desvantagens na proposta inicial de Slhoub, Carvalho e Bond (2017). A primeira é que somente a linguagem de comunicação seria insuficiente para especificar a comunicação dos agentes. A segunda é que especificar a forma de comunicação em seção diversa da linguagem pode dificultar a leitura do documento de especificação de requisitos. Para eliminar essas desvantagens estendemos a proposta de

Slhoub, Carvalho e Bond (2017), de forma a agruparmos a especificação da comunicação do sistema considerando tanto a troca de mensagens entre agentes quanto de agentes e usuários externos. Além disso, também adicionamos novas informações relacionadas a comunicação. Dessa forma, propomos a subseção 1.3.3 denominada de "Perspectiva de comunicação", ela foi subdividida em cinco subseções, a saber:

- Subseção 1.3.3.1 - "Forma de comunicação entre agentes e usuários externos";
- Subseção 1.3.3.2 - "Tipo de informação a ser transmitida entre os agentes e usuários externos";
- Subseção 1.3.3.3 - "Forma de comunicação entre os agentes";
- Subseção 1.3.3.4 - "Padrão de comunicação dos agentes"; e
- Subseção 1.3.3.5 - "Tipo de informação a ser transmitida entre os agentes".

A próxima subseção de nossa proposta é a 1.3.4, denominada de "Perspectiva do ambiente", nela é descrita a plataforma que o software será executado (como por exemplo *desktop, web ou mobile*). Além disso, também devem ser descritas a arquitetura dos agentes e papéis de agentes presentes no sistema.

Já a subseção 1.3.5 do padrão tradicional, que corresponde às "Funções do produto" foi estendida para cobrir as funcionalidades acessadas por agentes. Para isso, adicionamos duas subseções que correspondem à "Funcionalidades que podem ser acessadas por usuários normais" (subseção 1.3.5.1) e "Funcionalidades que podem ser acessadas por papéis de agentes" (subseção 1.3.5.2). Dessa forma, nesta subseção serão especificadas as funcionalidades acessadas tanto por usuários normais quanto por agentes.

Já a seção 3, do padrão tradicional foi totalmente reformulada para suportar características específicas de SMAs. Algumas subseções serão utilizadas tal e qual são propostas no padrão, mas outras seções foram adicionadas por nosso estudo, conforme detalhamos a seguir.

A subseção 3.9 corresponde à "Atribuição de papéis de agentes", nela os papéis de agentes serão descritos em nível de atribuições. Essas atribuições definem as responsabilidades de cada agente ao assumir determinado papel.

A subseção 3.10, corresponde à "Perspectiva de papéis de agentes", nela serão descritas as características internas dos agentes. Essa subseção é dividida em 7 subseções, conforme descrevemos a seguir:

- Subseção 3.10.1 - "Crenças iniciais do papel de agente";
- Subseção 3.10.2 - "Crenças necessárias para o objetivo se tornar uma intenção";
- Subseção 3.10.3 - "Percepções associadas ao papel de agente";

- Subseção 3.10.4 - "Crenças que podem ser afetadas pelas percepções";
- Subseção 3.10.5 - "Possíveis ações externas associadas ao objetivo";
- Subseção 3.10.6 - "Possíveis planos associados ao objetivo"; e
- Subseção 3.10.7 - "Possíveis ações externas associadas ao plano".

Já a subseção 3.11, corresponde à "Perspectiva de casos de uso internos". Nessa subseção se deve descrever cenários de casos de uso do tipo *Goal, Perception e Plan*. Um cenário ilustra um comportamento do agente (sequência de ações) enquanto ele pretende atingir um objetivo em um contexto específico (um estado específico do sistema), ou seja, mostra a realização do objetivo (CHOREN; LUCENA, 2005). Dessa forma, criamos três subseções, a saber:

- Subseção 3.11.1 - "Casos de uso internos do tipo *Goal*";
- Subseção 3.11.2 - "Casos de uso internos do tipo *Perception*"; e
- Subseção 3.11.3 - "Casos de uso internos do tipo *Plan*".

A subseção 3.12, corresponde à "Diagramas de casos de uso", nela se deve detalhar os diagramas de casos de uso e está dividida em duas subseções, a saber:

- Subseção 3.12.1 - "Diagrama de caso de uso para cada AgentRoleActor": nessa subseção se deve detalhar cada diagrama de casos de uso interno específico para cada um dos papéis de agentes. Os AgentRoleActors representam papéis assumidos por agentes que podem ter estereótipos do tipo ReactiveAgentRoles ou Cognitive-AgentRoles, esses papéis de agentes são modelados dentro da fronteira do sistema; e
- Subseção 3.12.2 - "Diagrama de caso de uso geral": esta subseção corresponde ao modelo de diagrama de casos de uso contendo todos os atores externos (usuários normais) que interagem com o sistem e os casos de uso que eles acessam. Além disso, nessa subseção também são representados os papéis que os agentes podem assumir, bem como os casos de uso internos acessados pelos agentes quando assumem um determinado papel.

A seção 4 que corresponde à "Verificação" no padrão tradicional, não estabelece de que forma o DERS deve ser verificado e validado, apenas menciona que deve ser executado em paralelo com a seção 3 (Requisitos). Também não é sugerida nenhuma técnica específica para inspeção do DERS. Dessa forma, criamos seis subseções para verificação do DERS utilizando a técnica de inspeção PBR adaptada para sistemas multiagentes. O Capítulo 4 apresenta mais detalhes de nossa técnica de inspeção. Estas seis subseções correspondem à:

- Subseção 4.1 - “Inspeção geral do documento de especificação de requisitos”;
- Subseção 4.2 - “Inspeção geral de cada caso de uso interno”;
- Subseção 4.3 - “Inspeção de cada caso de uso interno do tipo *Goal*”;
- Subseção 4.1 - “Inspeção de cada caso de uso interno do tipo *Perception*”;
- Subseção 4.5 - “Inspeção de cada diagrama de caso de uso parcial para cada AgentRoleActor”; e
- Subseção 4.6 - “Inspeção do diagrama de caso de uso geral”.

Na subseção 4.1 “Inspeção geral do documento de especificação de requisitos”, se deve inspecionar todo o DERS para localizar inconformidades, tanto relacionadas a características específicas de SMA quanto à estrutura geral do DERS. Já na subseção 4.2 “Inspeção geral de cada caso de uso interno”, o inspetor deve analisar cada caso de uso interno buscando possíveis inconsistências comuns de qualquer um deles (*Goal, Perception ou Plan*). Na subseção 4.3 “Inspeção de cada caso de uso interno do tipo *Goal*”, o inspetor deve analisar cada caso de uso interno do tipo *Goal* buscando inconformidades específicas desse tipo de caso de uso.

Já na subseção 4.4 “Inspeção de cada caso de uso interno do tipo *Perception*”, o inspetor deve analisar cada caso de uso interno do tipo *Perception* buscando inconformidades específicas desse tipo de caso de uso. Na subseção 4.5 “Inspeção de cada diagrama de caso de uso parcial para cada AgentRoleActor”, o inspetor deve analisar cada um dos diagramas de casos de uso parcial buscando inconformidades nos requisitos de corretude e consistência.

E por fim, a subseção “Inspeção do diagrama de caso de uso geral”, o inspetor deve analisar o diagrama de caso de uso geral buscando inconformidades no modelo que representa tanto os usuários normais, quanto os papéis de agentes.

5.2 Seções disponíveis no padrão utilizadas sem adaptações

Algumas seções que são propostas na especificação de requisitos do padrão ISO/IEC/IEEE 29148:2018 serão utilizadas em nosso estudo sem adaptações. Estas seções serão detalhadas a seguir.

Na subseção 1.1 denominada de “Propósito” se deve descrever a finalidade do software que esta sendo especificado. Já a subseção 1.3.1 que corresponde à “Perspectiva de produto” se deve definir a relação do sistema com outros produtos relacionados. Se o produto for um elemento de um sistema maior, se identifica as interfaces entre o produto coberto pela especificação de requisitos de software e o sistema maior do qual o produto é um elemento.

Já a subseção 1.3.3 do padrão tradicional que corresponde à "Características de usuários" foi renumerada para a subseção 1.3.6. No entando, deve ser utilizada da mesma forma que é proposta no padrão, ou seja, se deve descrever as características gerais dos grupos pretendidos de usuários do produto, incluindo características que podem influenciar a usabilidade, como nível educacional, experiência, deficiências e habilidade técnica.

Na subseção 1.3.7 que corresponde à "Limitações", devem ser descritas de forma geral quaisquer outros itens que irão limitar as opções do fornecedor, como por exemplo políticas e requisitos regulatórios, limitações de hardware e interfaces com outras aplicações.

Na subseção 1.4 "Definições", se deve fornecer as definições para quaisquer palavras ou frases que tenham um significado especial além do dicionário normal de definições. Dessa forma, utilizaremos esta subseção tal e qual é proposta no padrão tradicional.

A seção 2, corresponde às "Referências", e esta sendo utilizada tal e qual é proposta no padrão. Ela descreve todo e qualquer artefato utilizado para elaboração do documento de especificação de requisitos de software. Como por exemplo, um documento fornecido pelas partes envolvidas ou o próprio padrão ISO/IEC/IEEE 29148:2018 (ISO/IEC/IEEE..., 2018).

Na subseção "Funções" (3.1) do padrão tradicional, conforme apresentada na Figura 1, se define as ações fundamentais que devem ocorrer no software ao aceitar e processar as entradas. Também se deve considerar o processamento e geração de saídas. Isso inclui, entre outras informações, as verificações de validade de entrada, a sequência exata de operações, tratamento e recuperação de erros, e faltas e falhas de hardware.

A subseção "Requisitos de desempenho" (3.2) tem como objetivo detalha os requisitos não funcionais do tipo desempenho. Esses requisitos podem incluir o número de terminais a serem suportados, o número de usuários simultâneos a serem atendidos e a quantidade e o tipo de informação a tratar. Os requisitos de desempenho devem ser declarados em termos mensuráveis, como por exemplo, 95% das transações serão processadas em menos de 1 segundo.

A subseção de "Requisitos de usabilidade" (3.3) tem como propósito detalhar requisitos não funcionais do tipo usabilidade. Pode pode incluir eficácia mensurável, eficiência, critérios de satisfação e prevenção de danos que possam surgir em contextos específicos de uso.

A subseção de "Requisitos de interface" (3.4), tem como propósito especificar as interfaces do sistema com os usuários e com entidades externas. As entidades externas são equivalentes a outros sistemas. Nesta seção, também são definidas quaisquer interdependências ou restrições associadas às interfaces (por exemplo, protocolos de comunicação, dispositivos especiais, padrões).

Já a subseção de "Requisitos lógicos de banco de dados" (3.5) define qualquer informação a ser colocada em um banco de dados, incluindo, entidades de dados e seus

relacionamentos, restrições de integridade e segurança.

A subseção 3.6, denominada de "Restrições de projeto", é proposta pelo próprio padrão tradicional. Nesta seção, se deve especificar as restrições no projeto do sistema impostas por padrões externos, requisitos regulatórios ou limitações do projeto.

Na subseção "Atributos de sistemas e software" (3.7) do padrão tradicional, devem ser especificados atributos referente a requisitos não funcionais como por exemplo confiabilidade, disponibilidade e segurança. Dessa forma, utilizaremos esta seção tal e qual é proposta no padrão tradicional.

E na subseção "Informações de apoio" (3.8) do padrão tradicional, se deve detalhar informações adicionais de apoio tais como: a) os formatos de entrada e saída; b) os resultados de pesquisas com usuários; e c) uma descrição dos problemas a serem resolvidos pelo software. A última seção de nossa proposta de adaptação do padrão corresponde aos "Apêndices" (seção 5), esta seção será utilizada tal e qual é proposta no padrão tradicional. Ela contém duas subseções, a saber:

- Subseção 5.1 - "Suposições e dependências": nesta subseção, se deve listar todas as suposições e dependências aplicáveis aos requisitos que devem ser levadas em consideração na alocação e derivação dos requisitos de baixo nível [1]; e
- Subseção 5.2 - "Acrônimos e abreviações": nesta subseção se deve detalhar todas as siglas e abreviações utilizadas no documento de especificação de requisitos.

[1] Os requisitos de baixo nível ou *lower-level*, são frequentemente chamados de requisitos derivados. Requisitos derivados são originados do processo de decomposição de sistemas. São requisitos necessários para o sistema, mas não necessariamente fornecem um benefício direto para o usuário final. Ou podem ser requisitos de interface, que são os requisitos que surgem quando os subsistemas precisam se comunicar entre si para obter um resultado geral (TANG; HUANG, 2011). São exemplos de requisitos de baixo nível os cálculos no sistema (GANGADHARAN, 2020)(TANG; HUANG, 2011) e interface de comunicação entre sistemas (TANG; HUANG, 2011).

6 QUASE-EXPERIMENTOS

Desenvolvemos dois quase-experimentoss para validar a hipótese que a técnica de inspeção proposta neste estudo é eficaz na detecção de falhas, considerando que, segundo Wohlin (2012) os experimentos e quase-experimentos podem ser conduzidos para gerar, confirmar e estender teorias.

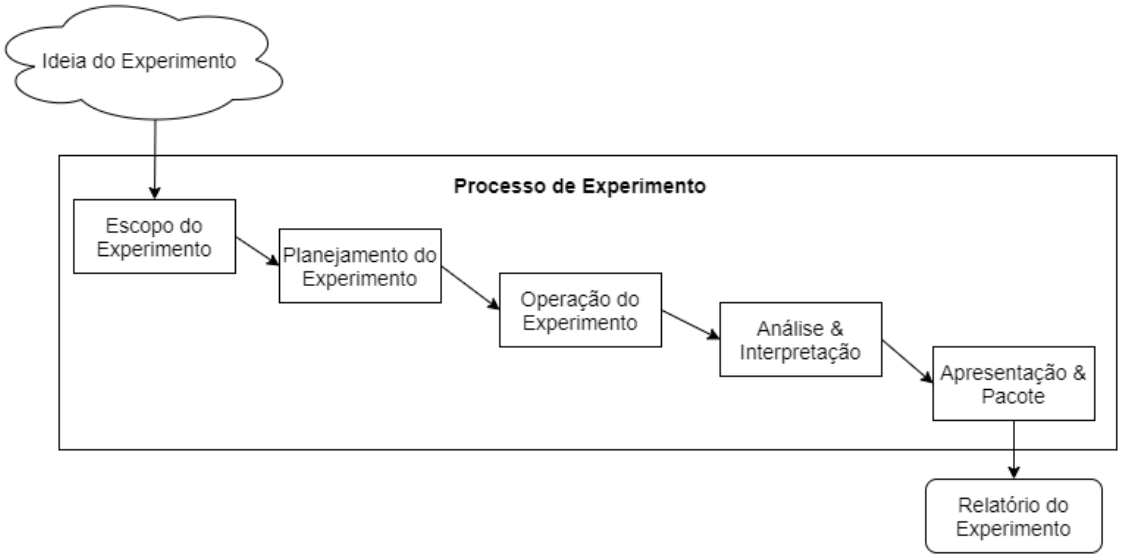
Segundo Wohlin et al. (2000), a força de um experimento ou quase-experimento está em sua capacidade de fornecer um contexto no qual certos padrões ou métodos são recomendados para uso. Dessa forma, poderemos avaliar se a técnica de inspeção proposta é adequada para inspeção de documentos de especificação de requisitos de SMAs.

Já de acordo com Juristo (2001), a experimentação está consolidada na detecção de mudanças quantificáveis como um meio de comparar um experimento ou quase-experimento com outro em busca de diferenças entre eles. Assim sendo, os dados coletados poderão ser caracterizamos por meio de análises estatísticas para comparar, entre os quase-experimentos, o desempenho individual e do conjunto de sujeitos.

As principais vantagens de um experimento ou quase-experimento são o controle (assuntos, objetos e instrumentação) e a capacidade de realizar análises estatísticas usando métodos de teste de hipótese. O controle garante que possamos estabelecer mais conclusões gerais (WOHLIN, 2012).

A execução de nossos quase-experimentos seguiram o processo definido por Wohlin (2012), conforme apresentamos na Figura 7.

Figura 7 – Visão geral do processo de experimento.



Fonte: Wohlin (2012).

6.1 Escopo

Analisar a técnica de inspeção proposta neste estudo para fins de avaliação no que diz respeito à eficácia, eficiência e aplicabilidade com relação à sua capacidade de detecção de falhas do ponto de vista do pesquisador no contexto de alunos de graduação em engenharia de software, alunos de mestrado e especialistas inspecionando um documento de especificação de requisitos de software de um sistema multiagente.

Os quase-experimentos foram conduzidos por dois pesquisadores, gerando resultados quanto à eficácia no desempenho da equipe de inspeção e à dificuldade em localizar diferentes classes de falhas utilizando a técnica de inspeção proposta.

6.2 Planejamento

A atividade de planejamento é onde a base para o experimento ou quase-experimento é lançada. Dessa forma, apresentamos a seguir mais detalhes sobre a preparação do material, a formulação das hipóteses, seleção de variáveis e os sujeitos selecionados.

6.2.1 O Documento de Especificação de Requisitos de Software

O documento usado nos quase-experimentos consiste em um Documento de Especificação de Requisitos de Software (DERS) com base no padrão ISO/IEC/IEEE 29148:2018 estendido para sistemas multiagentes, de acordo com a estrutura apresentada no Capítulo 5. O DERS descreve aspectos da segunda versão do sistema Heráclito. O sistema Heráclito é um sistema voltado ao aprendizado interativo e dialético com foco no ensino de dedução natural da lógica proposicional (GALAFASSI et al., 2017).

A extensão total do DERS finalizado é de 44 páginas. Entretanto, por questões de limitação de tempo para execução dos quase-experimentos, reduzimos o DERS a apenas 25 páginas. Dessa forma, foi possível descrever uma parte do sistema e inserir uma série de falhas que poderiam ser localizadas através do processo experimental. Para verificar a descrição do DERS entregue aos sujeitos participantes dos quase-experimentos consulte o ANEXO A.

6.2.2 Tipos de Falhas

Foram injetadas no documento de especificação de requisitos 25 falhas ou inconformidades de diferentes tipos. Essas falhas injetadas foram enquadradas em 4 tipos descritas a seguir. Utilizamos uma taxonomia composta por três letras após o nome do tipo de falha, cuja abreviatura será usada posteriormente nesta dissertação.

1. Incompleto (ICM) - Informações importantes da funcionalidade ou caso de uso interno omitidas no DERS;

2. Inconsistente (ICS) - Duas frases contidas no DERS se contradizem diretamente ou expressam ações que não podem ser corretas ou não podem ser executadas;

3. Incorreto (ICR) - Denota informação que não é verdadeira para as condições especificadas; e

4. Não rastreável (NRA) - As funcionalidades ou casos de uso internos não possuem identificadores únicos que permitam a rastreabilidade e histórico de alterações.

6.2.3 Hipóteses Formuladas

Neste estudo, buscamos examinar hipóteses sobre a eficácia da técnica de inspeção adaptada da PBR. Essas hipóteses estão relacionadas à adequação da proposta de inspeção para o contexto de sistemas multiagentes e à facilidade de aplicação da proposta de inspeção. Cada hipótese nula é seguida pela hipótese alternativa:

- Eficácia:

  QP-1: Foi possível detectar de forma eficaz falhas na especificação de requisitos?
  H0: O número total de falhas encontradas tem uma taxa inferior à 60%.
  H1: O número total de falhas encontradas tem uma taxa igual ou superior à 60%.

- Adequação:

QP-2: A proposta de inspeção de requisitos é adequada ao contexto de sistemas multiagentes?

H0: Mais de 60% dos sujeitos discordarão que a proposta de inspeção é adequada para o contexto de sistemas multiagentes.

H1: Mais de 60% dos sujeitos concordarão que a proposta de inspeção é adequada para o contexto de sistemas multiagentes.

- Facilidade de aplicação:

QP-3: A proposta de inspeção é de fácil aplicação?

H0: Mais de 60% dos sujeitos discordarão que a proposta de inspeção é de fácil aplicação.

H1: Mais de 60% dos sujeitos concordarão que a proposta de inspeção é de fácil aplicação.

A Hipótese alternativa (H1) que buscamos validar foi definida com um percentual de 60% considerando que, segundo Ackermann et al. (2007) uma inspeção de software aplicada adequadamente pode remover entre 60% a 90% de defeitos existentes. Dessa forma, definimos o limite inferior de 60% como uma hipótese válida.

6.2.4 Seleção de Variáveis

Segundo Wohlin (2012), antes que qualquer projeto possa começar, temos que escolher as variáveis dependentes e as independentes. Dessa forma, para este estudo empírico foram selecionadas duas variáveis dependentes, a saber: I) a primeira está relacionada a "Eficácia" da técnica de inspeção proposta na detecção de falhas em documentos de especificação de requisitos; e II) a segunda está relacionada a "Aplicabilidade" da técnica de inspeção proposta para o contexto de sistemas multiagentes. Por outro lado, as variáreis independentes estão relacionadas a: I) experiência dos sujeitos; e II) o número de falhas detectadas.

6.2.5 Os Participantes

Os participantes do primeiro quase-experimento foram 12 estudantes do curso de graduação em Engenharia de Software da Universidade Federal do Pampa. Todos os estudantes estavam matriculados na disciplina de Verificação e Validação. Esses alunos foram selecionados considerando que já estavam estudando conceitos e técnicas de verificação e validação de requisitos.

Já o segundo quase-experimento contou com 14 voluntários de diferentes perfis (6 alunos estudantes de mestrado, 1 com especialização, 2 com graduação e 5 alunos em fase de conclusão de curso), grande parte com experiência profissional na área de engenharia de software. Dos quatorze sujeitos participantes do segundo quase-experimento 3 tinham até 1 ano de experiência na área e 5 tinham mais de 2 anos de experiência, o que corresponde à aproximadamente 67% dos sujeitos com experiência profissional.

Para controlar as diferenças de conhecimento e habilidades entre os sujeitos, aplicamos um questinário prévio para identificar o perfil dos participantes. Cada sujeito preencheu um documento, chamado de Termo de Consentimento Livre e Esclarecido, que também coletou a autorização de participação de cada voluntário. Para consultar o Termo de Consentimento Livre e Esclarecido enviado aos sujeitos, verifique o ANEXO B.

Este documento coletou dados sobre o nível de conhecimento em sistemas multiagentes, o nível de conhecimento em especificação de requisitos, o nível de conhecimento em PBR e se o sujeito já inspecionou requisitos na academia ou indústria.

No primeiro quase-experimento, a maioria dos sujeitos se autoavaliaram com um conhecimento muito baixo em sistemas multiagentes. Para nivelar o conhecimento, preparamos um material sobre o contexto de sistemas multiagentes e apresentamos aos sujeitos no encontro de nivelamento.

Já no segundo quase-experimento, a maioria dos sujeitos se autoavaliaram com um conhecimento em sistemas multiagentes de médio para alto. Embora os sujeitos considerassem ter um bom conhecimento sobre SMA, também apresentamos conceitos sobre este tipo de sistema.

Em ambos os quase-experimentos grande parte dos sujeitos se autoavaliaram com um conhecimento de médio para alto em especificação de requisitos. Embora os sujeitos se autoavaliassem com um bom conhecimento neste contexto, também preparamos um material reforçando conceitos de especificação de requisitos.

Em ambos os quase-experimentos os sujeitos se autoavaliaram com um conhecimento de médio para baixo em Leitura Baseada em Perspectiva. Dessa forma, também preparamos um material com ênfase para esta técnica de inspeção. A maioria dos sujeitos responderam que tinham um conhecimento de médio para alto em inspeção de requisitos. No entanto, para nivelar o conhecimento e minimizar a falta de experiência, no dia do encontro para inspeção do documento de especificação de requisitos apresentamos aos sujeitos de forma geral o documento que seria inspecionado e os artefatos que seriam usados durante os quase-experimentos. Dessa forma, explicamos como conduzir o processo de inspeção, quais documentos deveriam ser preenchidos e em que momento.

6.2.6 Avaliação de Validade

Um aspecto importante da qualidade de um quase-experimentos é sua validade, mas existem algumas ameaças potenciais à validade. Wohlin (2012) classifica as ameaças em quatro tipos: validade interna, validade externa, validade de construção e validade de conclusão.

Durante a execução dos quase-experimentos estávamos expostos a uma série de ameaças. Sendo assim, discutiremos nesta subseção as principais ameaças identificadas.

- **Validade Interna:**

O viés de seleção é uma potencial ameaça considerando que existem diferenças entre grupos selecionados para inspeção, o que pode afetar o resultado observado. Esta ameaça não existe em nossos quase-experimentos, tendo em vista que, nenhum grupo é formado, todos os sujeitos completaram as inspeções de forma individual e independente.

Testes repetidos são uma possível ameaça. Para mitigar esta ameaça, ao final do primeiro quase-experimento não divulgamos quais falhas foram injetadas. Ainda, para o segundo quase-experimento selecionamos vonluntários já formados ou em fase de conclusão de curso com perfil diferente do primeiro. Além disso, substituímos algumas falhas injetadas por outras do mesmo grau de dificuldade.

Há o risco de que o histórico afete os resultados experimentais, uma vez que as circunstâncias não são exatamente as mesmas em ambas as ocasiões (WOHLIN, 2012). Para mitigar esta ameaça, o primeiro quase-experimento foi aplicado durante os horários de aulas da disciplina de Verificação e Validação. Já o segundo quase-experimento foi aplicado em horários diferenciados para cada sujeito ou grupo de sujeitos considerando a disponibilidade individual de cada um.

- **Validade Externa:**

Uma possível ameaça está relacionada à interação de seleção e tratamento. Segundo Wohlin (2012), esse é um efeito de ter uma população de sujeitos, não representativa da população para a qual queremos generalizar. Para mitigar esta ameaça, aplicamos um segundo quase-experimento em que foram selecionados sujeitos já formados ou em fase de conclusão de curso, sendo que a maioria possui experiência profissional na área de engenharia de software.

Outra ameaça diz respeito à interação de ambiente e tratamento. Segundo Wohlin (2012), esse é o efeito de não ter no quase-experimento cenário ou material representativo, por exemplo, da prática industrial. Para mitigar esta ameaça, utilizamos um documento de especificação de requisitos com base em uma adaptação do modelo (*template*) do padrão ISO/IEC/IEEE 29148:2018, em que se descrevia um sistema real (Heráclito). Esse padrão, segundo Franch et al. (2021) é utilizado na indústria para escrever requisitos. Além disso, em nossos quase-experimentos permitimos aos sujeitos aumentarem suas habilidades em inspeções, considerando que, segundo Anu et al. (2019) o conhecimento sobre inspeções de requisitos é uma habilidade chave da indústria. Em relação ao ambiente, como os quase-experimentos foram executados na modalidade de videoconferência, isto facilitou a separação dos sujeitos para que executassem as inspeções de forma individual e independente.

Uma terceira ameaça potencial é o tamanho do documento de especificação de requisitos ou das listas de verificação do GIRSMA. Se o tamanho for inadequado, pode afetar o resultado das inspeções. Para mitigar esta ameaça, reduzimos o documento de especificação do sistema Heráclito a apenas 25 páginas, em que descrevemos parte desse sistema. Em relação às listas de verificação, trabalhos adicionais são necessários para avaliar a capacidade de localizar as falhas injetadas em um tempo maior do que o disponibilizado em nossos quase-experimentos.

- **Validade de Construção:**

Uma das ameaças potenciais desse tipo que poderia ser aplicável aos nossos quase-experimentos é o viés de mono-operação. Para mitigar esta ameaça, nós aplicamos um segundo quase-experimento em que selecionamos voluntários com perfil diferente dos sujeitos do primeiro quase-experimento.

Outra possível ameaça aplicada em inspeções, segundo Wohlin (2012), é o viés de mono-método, onde a classificação de falhas é com base no julgamento subjetivo. Para mitigar esta ameaça, injetamos 25 falhas que poderiam ser localizadas com base nas listas de verificações do GIRSMA, sendo assim, evitamos a subjetividade e interpretação de falhas.

Uma possível ameaça está relacionada a adivinhação de hipóteses em que, segundo Wohlin (2012), se refere a quando as pessoas participantes de um experimento ou quase-experimento podem tentar descobrir qual é o propósito e o resultado pretendido do experimento ou quase-experimento. Para mitigar esta ameaça, não revelamos aos sujeitos a quantidade de falhas injetadas e informamos que o propósito do quase-experimento era avaliar a técnica de inspeção proposta por meio do GIRSMA.

A última ameaça de construção identificada está relacionada à apreensão de avaliação, que segundo Wohlin (2012) diz respeito ao medo de algumas pessoas em serem avaliadas. Para mitigar esta ameaça, reforçamos a todo o momento que o quase-experimento tinha como objetivo avaliar a técnica de inspeção proposta e não avaliar o desempenho individual de cada sujeito.

- **Validade de Conclusão:**

A confiabilidade da implementação do tratamento poderia ficar comprometida caso algum sujeito não entendesse a forma de aplicar a inspeção ou como preencher o formulário de inconformidade. Para mitigar esta ameaça, durante o encontro de nivelamento, nós explicamos cuidadosamente como utilizar o GIRSMA e em que momento preencher o formulário de inconformidades.

Considerando que o quase-experimento foi aplicado na modalidade de videoconferência, poderiam ocorrer irrelevâncias aleatórias no cenário experimental, como ruídos fora do quase-experimento ou interrupção de sua execução. Para mitigar esta ameaça, solicitamos que os sujeitos permanecessem logados na vídeo chamada e em vários momentos questionamos se haviam dúvidas na execução do quase-experimento. Dessa forma, tentamos manter o foco e comprometimento dos sujeitos.

A heterogeneidade aleatória de sujeitos é comum em grupos de estudos. Para mitigar esta ameaça, definimos um dia como encontro de nivelamento. Dessa forma, apresentamos aos sujeitos conceitos utilizados no quase-experimento com ênfase nos assuntos que eles se autoavaliaram com baixo nível de conhecimento. Sendo assim, poderíamos garantir um grupo de sujeitos homogêneo.

6.3 Operação

Esta atividade envolve a preparação dos assuntos, bem como o material necessário para execução do quase-experimento. Além disso, envolve a execução conforme planejamento e validação dos dados. A seguir apresentamos como realizamos cada uma dessas etapas em nossos quase-experimentos.

6.3.1 Preparação

Esses quase-experimentos foram os primeiros estudos empíricos com o objetivo de avaliar nossa proposta de inspeção de documentos de especificação de requisitos para sistemas multiagentes. Essa avaliação considerou a eficácia da técnica de inspeção na detecção de falhas no documento de especificação de requisitos, sua adequação ao contexto de sistemas multiagentes e sua aplicabilidade.

O primeiro quase-experimento foi conduzido no mês de março de 2021. Já o segundo quase-experimento foi conduzido entre os meses de julho e agosto de 2021. Em cada quase-experimento foi realizado três encontros todos on-line se utilizando a videoconferência para comunicação com os voluntários.

No primeiro encontro, apresentamos o contexto de aplicação desse estudo aos candidatos à voluntários. Nesse encontro, os candidatos foram convidados a preencher o formulário "Termo de Consentimento Livre e Esclarecido", em que aceitavam participar do quase-experimento.

O segundo encontro, teve como objetivo nivelar os conhecimentos sobre conceitos relacionados ao contexto do quase-experimento, em que se apresentou o documento de especificação desenvolvido e a técnica de inspeção criada para examiná-lo. No material apresentado aos alunos, enfatizamos conceitos a respeito dos quais os voluntários declararam ter baixo conhecimento, procurando assim ter um conjunto homogêneo de sujeitos.

Por fim, no terceiro encontro foi executada a inspeção. Inicialmente, distribuímos aos sujeitos o Documento de Especificação de Requisitos do sistema Heráclito com base no padrão ISO/IEC/IEEE 29148:2018 estendido, o Guia de Inspeção de Requisitos para Sistemas Multiagentes (GIRSMA) necessário à inspeção e o formulário para registrar as possíveis inconformidades encontradas.

Nos primeiros 30 minutos, explicamos cuidadosamente cada documento entregue. Nesse momento também explicamos como deveria ser conduzida a inspeção do DERS e como preencher o Formulário de Inconformidades. As 2 horas e 30 minutos seguintes, foram utilizadas pelos sujeitos para inspeção do DERS. Durante a execução da inspeção os sujeitos permaneceram logados na videoconferência e os pesquisadores ficaram *on-line* para esclarecer eventuais dúvidas durante a condução do quase-experimento.

Os sujeitos que concluíram o quase-experimento enviaram o Formulário de Inconformidades preenchido através de um formulário eletrônico, no qual também responderam

perguntas sobre a experiência com esse estudo empírico.

6.3.2 Execução

Cada sujeito participante do quase-experimento recebeu uma cópia do Documento de Especificação de Requisitos de 25 páginas, junto com um GIRSMA considerando a perspectiva do agente, denominado de PBR-Simulador de Agente e um Formulário de Inconformidades.

O GIRSMA continha instruções para condução da inspeção e os passos a serem seguidos durante a sua execução. Para consultar o GIRSMA entregue aos sujeitos, verifique o ANEXO C.

Se o sujeito acreditasse ter encontrado uma falha ou problema que exigisse atenção da equipe de requisitos, esta deveria ser anotada no Formulário de Inconformidades. Esse formulário descreve, entre outras informações, a localização e o tipo da falha. Para consultar o modelo de formulário entregue aos sujeitos, verifique o ANEXO D. Ao final da execução do quase-experimento, cada sujeito enviou o Formulário de Inconformidades preenchido por meio de um questionário eletrônico.

O tempo total disponibilizado, desde a distribuição do DERS até a entrega final do Formulário de Inconformidades preenchido pelos sujeitos foi de 3 horas. As falhas relatadas foram então analisadas. Se elas pertenciam a uma das 25 falhas que estávamos rastreando, então esta inconformidade foi inserida na planilha de tabulação dos dados. Se um erro não injetado foi relatado, analisamos cuidadosamente e caso aceito, ele entrou em nosso banco de erros. Com estas informações tabuladas geramos os resultados.

6.3.3 Validação dos dados

Durante a entrega dos materiais necessários para execução do quase-experimento, explicamos cuidadosamente como a inspeção deveria ser realizada e como preencher o Formulário de Inconformidades. Dessa forma, poderíamos garantir a confiabilidade dos dados inseridos pelos sujeitos. Além disso, durante a execução das inspeções os pesquisadores permaneceram logados na videoconferência para sanar eventuais dúvidas.

Para cada entrega do formulário de inconformidades, os pesquisadores conferiram se as informações estavam corretas. Caso o formulário não estivesse de acordo, os dados deveriam ser descartados.

6.4 Análise e interpretação

Os dados coletados na seção 6.3 (Operação), são dados de entrada para a análise e interpretação. Portanto, as falhas descobertas por cada um dos sujeitos foram inseridas em uma planilha de tabulação dos dados.

Dessa forma, foi possível analisar quantitativamente o número de falhas descobertas por cada sujeito. Além disso, caracterizamos os dados por meio de análises estatísticas do desempenho individual e do conjunto de sujeitos.

Como os perfis dos sujeitos era diferente de um quase-experimento para o outro, com base na interpretação dos dados coletados foi possível analisar e comparar se a experiência dos sujeitos influenciou os resultados.

6.5 Apresentação

Esta última atividade envolve a apresentação e documentação dos resultados. Portanto, com base nas principais descobertas com a interpretação dos dados discutiremos a seguir cada uma das questões de pesquisa formuladas.

6.5.1 Respondendo às questões de pesquisa

Os resultados mais importantes de nossos quase-experimentos são aqueles que respondem às questões levantadas na Subseção 6.2.3. Dessa forma, discutimos a seguir os resultados obtidos com a execução dos quase-experimentos.

- QP-1: Foi possível detectar de forma eficaz falhas na especificação de requisitos?

Para analisar a capacidade de detectar falhas usando a adaptação da técnica de leitura baseada em perspectiva apresentada nesse estudo, demonstraremos a seguir a quantidade de falhas descobertas em cada um dos quase-experimentos.

Dessa forma, apresentamos na Tabela 16 a quantidade de falhas detectadas por cada sujeito no primeiro quase-experimento. Para não relacionar os resultados com os sujeitos, utilizaremos um número identificador que contém a letra “S” (Sujeito) seguido do número atribuído na planilha de tabulação dos dados.

Conforme podemos observar na Tabela 16, os doze sujeitos do primeiro quase-experimento encontraram em média 33,67% das 25 falhas injetadas no DERS. Também observamos que apenas, dois sujeitos encontraram mais de 50% das falhas injetadas, ou seja, um localizou 52% e outro 60%. O sujeito que reportou o maior número de falhas registrou 15 falhas das 25 injetadas, o que corresponde a 60%. Esse percentual está dentro da expectativa definida por Ackermann et al. (2007), de que uma inspeção de software aplicada adequadamente pode remover entre 60% a 90% de defeitos existentes.

Também observamos que das 25 falhas injetadas no DERS, apenas duas não foram reportadas por nenhum dos doze sujeitos, isso corresponde a 8% das falhas presentes no DERS. As duas falhas não localizadas são:

- Inexistência de Cenário Alternativo no Caso de Uso Interno ou *Internal Use Case* (IUC) “Informar erro em questão ao aluno”; e

Tabela 16 – Falhas encontradas e tempo gasto na inspeção por sujeito no primeiro quase-experimento.

Sujeito	**Número de Falhas Detectadas**	**%**	**Tempo (HH:MM)**
S1	15	60	2:00
S2	08	32	2:20
S3	05	20	1:00
S4	09	36	1:40
S5	09	36	2:00
S6	05	20	2:00
S7	04	16	2:09
S8	07	28	1:30
S9	12	48	1:00
S10	06	24	1:30
S11	13	52	1:30
S12	08	32	1:30
N	25	100	2:30
Média	8,42	33,67	1:35

Fonte: Autor.

- Inexistência do IUC "Perceber perfil do aluno".

Consideramos que não localizar a falha "Inexistência de cenário alternativo" no IUC "Informar erro em questão ao aluno", é totalmente aceitável considerando que nem todo Caso de Uso Interno necessita de cenário alternativo. Julgar com propriedade a necessidade de cenário alternativo requer um conhecimento avançado em sistemas multiagentes e experiência com o escopo do sistema especificado. Apenas com o encontro de nivelamento não poderíamos garantir um aumento suficiente no nível de conhecimento em sistemas multiagentes e no escopo do sistema Heráclito.

Já a falha de "Inexistência" do IUC "Perceber perfil do aluno", podemos considerar que houve uma falta de atenção dos inspetores. Levando em consideração que, o caso de uso interno "Perceber perfil do aluno" é executado no cenário principal dos casos de uso internos "Selecionar questão na base de conhecimento" e "Determinar conteúdo", mas o DERS não contém o caso de uso interno "Perceber perfil do aluno" descrito separadamente. Dessa forma, deveria ter sido apontada uma falha no IUC "Perceber perfil do aluno".

Embora a taxa média de detecção de falhas tenha sido baixa, consideramos que a descoberta de 23 falhas das 25 injetadas é um bom indício da eficácia da técnica de inspeção proposta nesse estudo. Observamos que algumas falhas injetadas não foram localizadas por falta de atenção dos inspetores. Por exemplo, falhas básicas como a falta de cenário principal nos casos de uso internos. Este tipo de falha foi localizada por apenas 4 inspetores nos casos de uso internos "Perceber solicitação de nova questão pelo

Mediador" e "Preparar aplicação da questão". O que nos chamou a atenção é que apenas dois inspetores registraram este tipo de falha no caso de uso interno "Verificar questão do aluno". Por outro lado, o caso de uso interno "Verificar ação do aluno" do mesmo tipo dos anteriores obteve um número considerado aceitável, totalizando 8 inspetores registrando esta falha.

Dessa forma, supomos que houve uma falta de atenção dos inspetores, considerando que esta falha é classificada como de nível baixo de dificuldade e esperávamos um número maior de inspetores localizando este tipo de falha.

Os inspetores eram todos alunos de graduação da disciplina de Verificação e Validação que tinham apenas recentemente aprendido sobre inspeções, principalmente a técnica PBR. Além disso, eles tinham experiência em inspeção apenas com um trabalho prático aplicado naquela disciplina. Os conhecimentos em modelagem e documentação de caso de uso foram adquiridos em um semestre anterior ao nosso quase-experimento. No entanto, os sujeitos selecionados tinham um certo conhecimento sobre os assuntos abordados em nosso estudo.

Já os resultados obtidos com o segundo quase-experimento são demonstrados na Tabela 17.

Tabela 17 – Falhas encontradas e tempo gasto na inspeção por sujeito no segundo quase-experimento.

Sujeito	**Número de Falhas Detectadas**	**%**	**Tempo (HH:MM)**
S13	10	40	1:40
S14	15	60	2:56
S15	13	52	1:30
S16	07	28	2:00
S17	15	60	2:00
S18	07	28	2:10
S19	21	84	2:15
S20	17	68	2:27
S21	05	20	3:00
S22	11	44	2:34
S23	10	40	1:49
S24	17	68	1:30
S25	09	36	2:17
S26	16	64	1:55
N	25	100	2:30
Média	12,36	49,43	2:05

Fonte: Autor.

Conforme podemos observar na Tabela 17, os quatorze sujeitos do segundo quase-experimento encontraram em média 49,43% das 25 falhas injetadas no DERS. Comparando esse resultado com o primeiro quase-experimento observamos que houve um au-

mento na média de falhas detectadas. Além disso, podemos verificar que sete sujeitos encontraram mais de 50% das falhas injetadas, ao contrário do primeiro quase-experimento que apenas dois atingiram esse percentual.

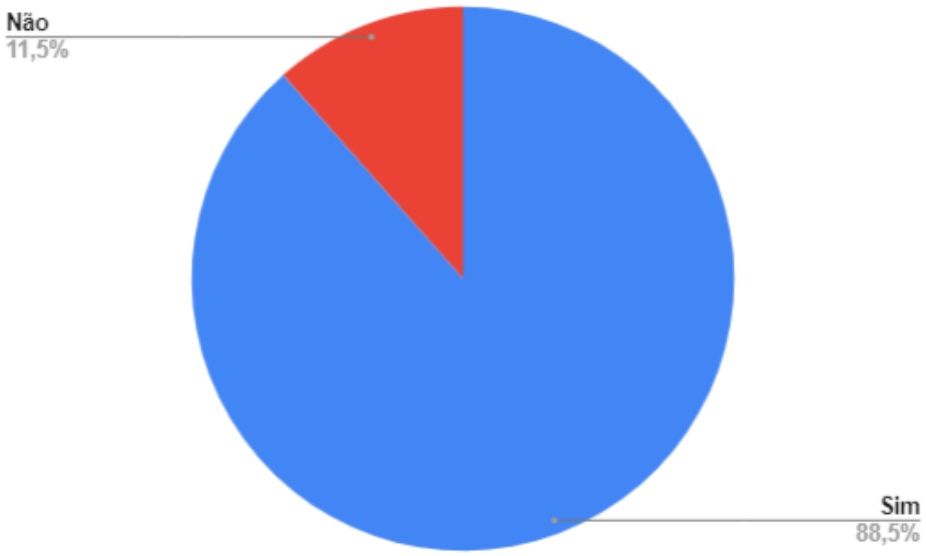
Nesse segundo quase-experimento seis sujeitos localizaram 60% ou mais das falhas injetadas, ao contrário do primeiro quase-experimento que apenas um localizou esse percentual. Desses seis sujeitos o maior percentual corresponde a 84% das falhas injetadas sendo detectadas.

Também observamos que no segundo quase-experimento 100% das falhas injetadas foram localizadas por pelo menos um dos sujeitos, ao contrário do primeiro que 2 falhas não foram localizadas. Com base nesses resultados, observamos que a experiência dos sujeitos pode ter influenciado os resultados da inspeção, considerando que esse segundo quase-experimento contou com alunos já formados ou em fase de conclusão de curso.

- QP-2: A proposta de inspeção de requisitos é adequada ao contexto de sistemas multiagentes?

Para responder a esta questão de pesquisa, solicitamos a opinião dos sujeitos por meio de um questionário respondido ao final da execução do quase-experimento. Entre os 26 sujeitos participantes o total de 23 concordaram que o GIRSMA é adequado para o contexto de SMAs, o que corresponde a 88,5% dos sujeitos, conforme pode ser observado na Figura 8.

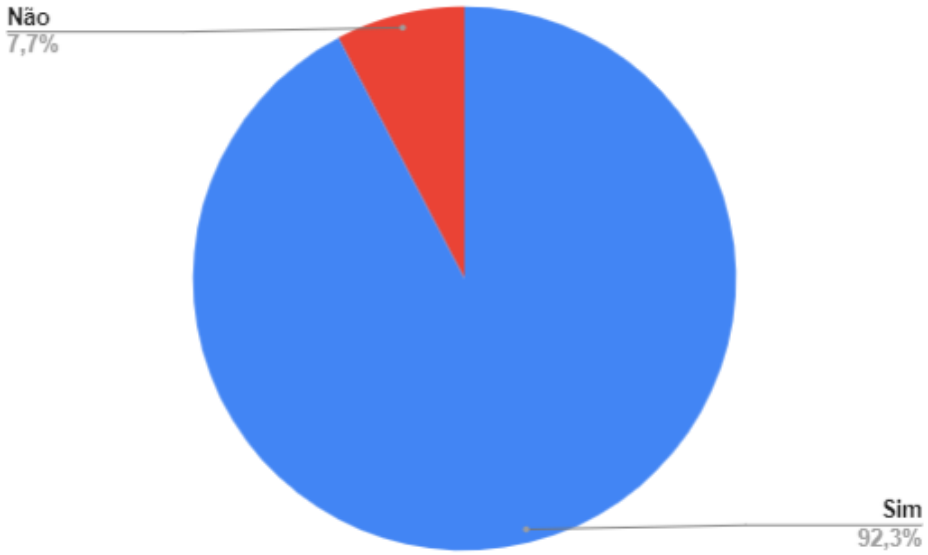
Figura 8 – Taxa de concordância de que o GIRSMA é adequado para SMAs.



Fonte: Autor.

Já o total de 24 sujeitos concordaram totalmente que as listas de verificações do GIRSMA ajudaram a detectar falhas no documento de especificação de requisitos, o que corresponde a 92,3% dos sujeitos, conforme pode ser observado na Figura 9.

Figura 9 – Taxa de concordância de que as listas de verificações auxiliaram na inspeção de SMA.



Fonte: Autor.

- QP-3: A proposta de inspeção é de fácil aplicação?

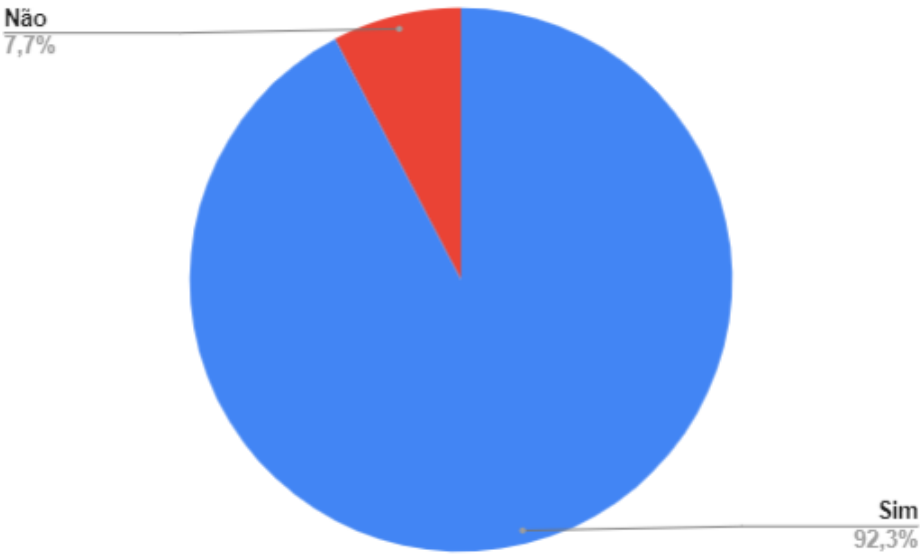
Para responder a esta questão de pesquisa, solicitamos a opinião dos sujeitos por meio de um questionário respondido ao final da execução do quase-experimento. No total 92,3% dos sujeitos responderam que a proposta de inspeção é fácil de usar, conforme pode ser observado na Figura 10. Já os sujeitos que responderam discordar que a proposta é fácil de usar, totalizou 7,7%. Dessa forma, obtivemos um percentual superior a 60% de concordância.

Já os sujeitos que responderam concordar que a proposta de inspeção é fácil de seguir, totalizou 88,5% dos sujeitos, conforme pode ser observado na Figura 11. E um total de 11,5% dos sujeitos responderam discordar que a proposta é fácil de seguir. Dessa forma, obtivemos um percentual superior a 60% de concordância.

6.5.2 Classificação dos Tipos de Falhas por Dificuldade de Detecção

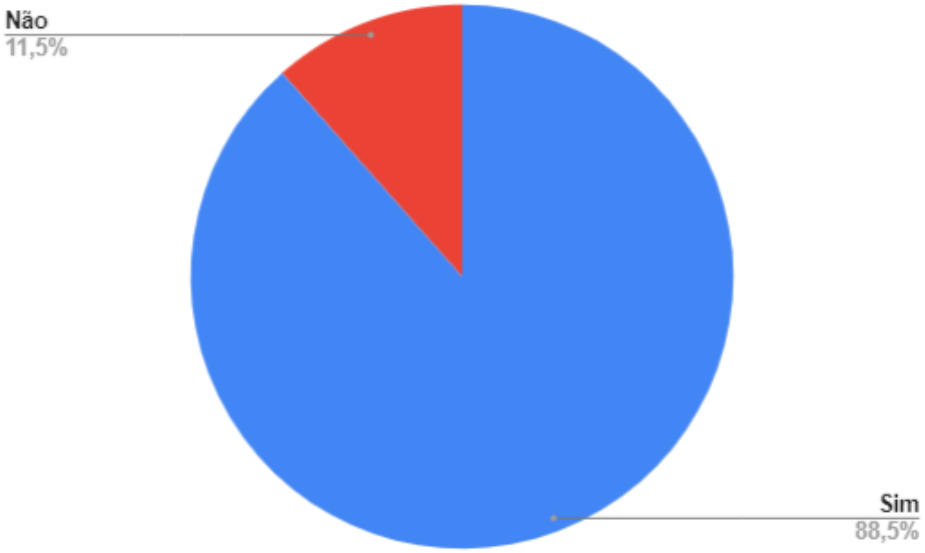
Para entender a dificuldade dos sujeitos em localizar um dos quatro tipos de falhas definidas na subseção 6.2.2, apresentamos na Tabela 18 a frequência com que os sujeitos encontraram esses tipos de falhas no primeiro quase-experimento. Como pode ser observado, uma média de 3,12 das falhas do tipo ICS foram localizadas, isso corresponde a uma taxa média de 26% dos doze sujeitos encontrando este tipo de falha. No entanto, eles foram capazes de encontrar em média 4,42 de 7 falhas injetadas do tipo ICR, cerca de 36,83% dos sujeitos localizando este tipo de falha.

Figura 10 – Taxa de concordância de que a técnica proposta é fácil de usar.



Fonte: Autor.

Figura 11 – Taxa de concordância de que a técnica proposta é fácil de seguir.



Fonte: Autor.

Já no segundo quase-experimento a taxa de detecção de falhas teve um percentual melhor comparado ao primeiro quase-experimento, conforme resultados apresentamos na Tabela 19.

Conforme pode ser observado na Tabela 19, as falhas do tipo ICM foram detectadas em média por 8,28 dos 14 sujeitos participantes do segundo quase-experimento, isso corresponde a 59,14% contra os 33,33% do primeiro quase-experimento. Analisando individualmente este tipo de falha observamos que 3 das falhas injetadas do tipo ICM

Tabela 18 – Detecção de falhas por tipo de falha no primeiro quase-experimento.

Tipo de Falha	Total Injetada	Número Mínimo	Número Máximo	Média de Falhas Detectadas	Taxa Média de Detecção
ICM	9	0	8	4,00	33,33%
ICR	7	2	8	4,42	36,83%
ICS	8	0	8	3,12	26,00%
NRA	1	9	9	9,00	75,00%

Fonte: Autor.

Tabela 19 – Detecção de falhas por tipo de falha no segundo quase-experimento.

Tipo de Falha	Total Injetada	Número Mínimo	Número Máximo	Média de Falhas Detectadas	Taxa Média de Detecção
ICM	7	1	12	8,28	59,14%
ICR	9	1	9	5,77	41,21%
ICS	8	2	12	6,62	47,29%
NRA	1	12	12	12	85,71%

Fonte: Autor.

foram localizadas por 12 sujeitos, isso corresponde a 85,71% dos sujeitos registrando este tipo de falha.

Já das 9 falhas injetadas do tipo ICR em média 5,77 dos 14 sujeitos localizaram este tipo de falha, o que corresponde a 41,21%. Todavia, observamos que 3 das falhas injetadas do tipo ICR foram localizadas por 9 sujeitos, isso corresponde a 64,28%.

A falha do tipo ICS foi localizada em média por 6,62 dos 14 sujeitos, isso corresponde a 47,29%. No entanto, observamos que 2 falhas deste tipo foram localizadas por 12 dos 14 sujeitos, correspondendo a 85,71%. Por fim, a falha do tipo NRA foi localizada por 12 dos 14 sujeitos, isso corresponde a uma taxa de detecção de 85,71%.

De forma geral, observamos que no primeiro quase-experimento a falha do tipo Inconsistente (ICS) foi a mais difícil de localizar atingindo um percentual de 26% de taxa de detecção. No entanto, no segundo quase-experimento observamos que a falha do tipo Incorreta (ICR) foi a mais difícil de localizar com um percentual de 41,21% de taxa média de detecção.

Esses dados sugerem que alguns tipos de falhas podem ser mais difícies de localizar do que outros. Dessa forma, mais esforços devem ser colocados para localizar esses tipos de erros durante a inspeção do DERS.

6.6 Considerações finais

Estudos anteriores, como por exemplo de Schneider, Martin e Tsai (1992), Maldonado, Carver e Shull (2006), Liu et al. (2012) e Shan et al. (2015), apresentaram resultados semelhantes aos nossos para detecção de falhas em documentos de requisitos. No estudo

de Schneider, Martin e Tsai (1992), os inspetores localizaram em média 35,1% das falhas injetadas. Os participantes deste experimento foram 27 estudantes do curso de graduação em engenharia de software com diferentes níveis de experiências. A diferença deste estudo para os nossos quase-experimentos é que a execução da inspeção deste experimento foi realizada por um conjunto de 9 equipes de inspetores, já os nossos foram executados por inspeções individuais.

Já o estudo apresentado por Maldonado, Carver e Shull (2006), analisou a eficácia do uso da PBR para detecção de falhas em dois documentos de requisitos. Os sujeitos eram desenvolvedores de software profissionais do *Goddard Space* da *NASA Flight Center*. No primeiro documento os sujeitos localizaram 43,75% das falhas injetadas e no segundo documento foi localizado 56,8% das falhas.

O estudo apresentado por Liu et al. (2012), comparou a técnica PBR com um método de inspeção proposto pelos autores que visa determinar se cada cenário funcional definido na especificação é implementado corretamente. Foram selecionados 74 estudantes do segundo ano do curso de Ciência da Computação. Todos os sujeitos tinham experiência com inspeções apenas em ambiente acadêmico. No entanto, antes do experimento não tinham aplicado na prática a inspeção com uso da PBR. Dessa forma, os sujeitos localizaram 27% das falhas injetadas utilizando a técnica PBR e localizaram 35% das falhas utilizando o método proposto no estudo dos autores.

Já o estudo apresentado por Shan et al. (2015), avaliou a eficácia da técnica de inspeção na captura de 30 falhas injetadas em um documento de requisitos. O estudo de caso foi aplicado em um conjunto de 57 alunos do segundo ano de Engenharia de Software. Assim como em nosso primeiro quase-experimento, os alunos participantes do estudo de Shan et al. (2015) também tinham pouca experiência com inspeção de software. Dessa forma, a melhor detecção individual foi de 8 defeitos, o que corresponde à 26,67% dos defeitos injetados. Houve alunos que detectaram apenas 1 defeito injetado.

Embora o número médio de falhas encontradas pelos sujeitos em nosso primeiro quase-experimento tenha sido de 33,67%, verificamos que um total de 23 falhas injetadas no DERS, ou cerca de 92%, foram descobertas por pelo menos um dos doze sujeitos que realizaram a inspeção. Dessa forma, em nosso primeiro quase-experimento, a taxa de detecção de falhas foi elevada de 33,67% em N=1 para 92% em N=12. Acreditamos que isso seja uma evidência da eficácia da técnica de inspeção de requisitos proposta em nosso estudo e apresentada no Capítulo 4.

Outro ponto interessante, é que apenas 2 falhas das 25 presentes no DERS, cerca de 8%, não foram localizadas por nenhum sujeito no primeiro quase-experimento. Este resultado é bem melhor que o apresentado por Schneider, Martin e Tsai (1992), em que cerca de 23% das falhas injetadas não foram localizadas.

A falta de experiência dos sujeitos em inspeção de requisitos, pode ter influenciado o baixo desempenho individual no primeiro quase-experimento. Neste sentido, um estudo

apresentado por Sauer et al. (2000) destaca alguns fatores que podem influenciar a eficácia das técnicas de revisões de software, tais como:

- O nível de habilidade dos revisores individuais;
- Treinamento para melhorar a competência dos indivíduos; e
- Seleção de revisores que são especialistas em detecção de defeitos.

Por outro lado, o segundo quase-experimento teve um número médio de detecção de falhas de 49,43%. O desempenho individual também foi melhor considerando que, seis sujeitos localizaram mais de 60% das falhas injetadas. Além disso, todas as falhas foram localizadas por pelo menos um dos sujeitos. Dessa forma, observamos que a experiência dos inspetores teve influência nos resultados obtidos, uma vez que, para o segundo quase-experimento foram selecionados sujeitos já formados ou em fase de conclusão de curso e a maioria com experiência profissional na área de engenharia de software.

Como a leitura individual de artefatos depende de atributos do leitor, como o histórico educacional e experiência do inspetor (CARVER; NAGAPPAN; PAGE, 2008), os resultados obtidos com os dois quase-experimentos comprovam esta afirmação, considerando que o segundo quase-experimento obteve um resultado melhor que o primeiro.

Ao revisarmos o processo de inspeção após a conclusão do primeiro quase-experimento, verificamos que uma questão da Lista de Verificação apresentada na Tabela 15 e entregue aos sujeitos estava incorreta. A pergunta: "Cada percepção (Perception) está associada a pelo menos um caso de uso interno do tipo «Goal»?", poderia inserir um falso positivo, considerando que, se uma percepção estiver associada somente a um Plano, estará correto mas não vai satisfazer este requisito. Como solução, eliminamos esta questão, pois entendemos que poderia inserir um falso positivo. Dessa forma, foi possível eliminar tal erro antes da execução do segundo quase-experimento.

Os quase-experimentos também contribuíram para aumentar a experiência em inspeções de software dos alunos da disciplina de Verificação e Validação do curso de Engenharia de Software da Universidade Federal do Pampa e dos sujeitos já formados, considerando que, segundo Gazerani et al. (2015), muitos engenheiros de software sofrem com a falta de conhecimento prévio e experiência em inspeções de software e suas técnicas.

7 CONCLUSÃO

A verificação é uma fase necessária da engenharia de requisitos e desempenha um papel crucial para que qualquer projeto seja bem sucedido (GUPTA; DERAMAN; SIDDIQUI, 2019). Desta forma, analisamos todos os 54 estudos remanescentes da execução de nossa revisão sistemática da literatura a fim de determinar de que forma a subárea de validação estava sendo tratada e quais as técnicas de verificação e validação eram utilizadas. Deste modo, observamos que nenhum estudo apresentava uma subárea de verificação e validação de forma consistente e nenhum apresentava uma técnica específica para verificar e validar os requisitos produzidos.

Assim, nós propomos neste trabalho uma técnica de inspeção para verificar os Documentos de Especificação de Requisitos de Software (DERS) produzidos com base na extensão do modelo (*template*) do padrão ISO/IEC/IEEE 29148:2018 (ISO/IEC/IEEE..., 2018) e na notação MASRML (GUEDES et al., 2020). Para desenvolver esta técnica de inspeção, adaptamos a técnica de Leitura Baseada em Perspectiva (PBR) (BASILI et al., 1996; SHULL; RUS; BASILI, 2000), criando a perspectiva do Simulador de Agente.

Neste sentido, segundo Ciolkowski et al. (1997), a PBR deve ser adaptável ao documento particular (por exemplo, documento de requisitos) e à notação em que o documento está escrito. Além disso, qualquer técnica incluindo a PBR deve ser adaptada às particularidades de um determinado ambiente para ter mais sucesso. Dessa forma, o Guia de Inspeção de Requisitos para Sistemas Multiagentes (GIRSMA) proposto neste estudo constitui uma nova abordagem para verificar requisitos para este tipo de sistema.

Para aplicar a nossa proposta de inspeção produzimos um documento de especificação de requisitos de software. Este documento foi produzido com base em uma adaptação do *template* do padrão ISO/IEC/IEEE 29148:2018 para suportar à identificação de características necessárias ao modelo BDI. A adaptação proposta compreendeu a extensão do modelo deste padrão, em que adicionamos novas seções e renumeramos algumas seções sugeridas pelo próprio padrão para que os requisitos para SMA sejam especificados.

Nossa proposta de especificação de requisitos permite identificar as características necessárias ao modelo BDI, diferente do estudo apresentado por Slhoub, Carvalho e Bond (2017) que não suporta este tipo de representação. Utilizamos a descrição de cenários internos dos agentes tendo em vista que, segundo Cossentino et al. (2014a) cenários tem como objetivo refinar e explorar as possíveis sequências de interações entre objetivos (*Goals*). Desta forma, é possível obter uma descrição completa de cada caso de uso interno com base na sequência de passos, cenários principais, cenários alternativos e, se necessário as pré-condições envolvidas.

Além disso, ao utilizar a representação de casos de uso internos com base na MASRML (GUEDES et al., 2020) permitimos uma especificação de requisitos com base em uma notação adaptada para sistemas multiagentes. Diferente de estudos anteriores que utilizam a UML padrão para representação de requistos de SMAs.

Com base nos cenários de casos de uso internos do tipo objetivo (*Goal*) é possível especificar as percepções associadas ao objetivos, além de também permitir estabelecer quando as crenças viram intenções, diferentemente dos demais estudos que não suportam este tipo de representação.

Ainda, nossa proposta de especificação permite descrever os requisitos específicos de sistemas multiagentes sem deixar de lado a especificação de requisitos acessados por usuários normais. Isto diferencia nosso estudo dos demais tendo em vista que, outros estudos sobre especificação de requisitos para SMAs apresentam modelos que permitem apenas identificar requisitos acessados por agentes.

Com objetivo de avaliar a proposta de extensão do padrão, nós a utilizamos para especificar a segunda versão do sistema Heráclito (GALAFASSI et al., 2019), um sistema multiagente que visa auxiliar no ensino de lógica. Desta forma, esperamos que a especificação dos requisitos apresentada neste estudo possa auxiliar no desenvolvimento deste projeto.

Avaliamos a aplicabilidade e eficiência de nossa adaptação da técnica PBR, por meio de dois quase-experimentos. O primeiro quase-experimento contou com alunos da disciplina de Verificação e Validação do curso de Engenharia de Software da Universidade Federal do Pampa. Como resultado deste primeiro quase-experimento observamos que 23 falhas das 25 injetadas no DERS, ou cerca de 92%, foram descobertas por pelo menos um dos doze sujeitos que realizaram a inspeção.

Avaliando individualmente o desempenho dos sujeitos, observamos que a média de falhas encontradas foi de 33,67%. Este número é bem próximo de estudos anteriores de experimentos com inspeções de software, como por exemplo o estudo dos autores Shan et al. (2015), no qual o sujeito com melhor desempenho localizou 26,67% dos defeitos injetados.

Já no segundo quase-experimento foram selecionados voluntários já formados ou em fase de conclusão de curso e alguns com experiência profissional na área de engenharia de software. Como resultado deste segundo quase-experimento observamos que em média 49,43% das falhas foram localizadas. Além disso, 100% das falhas injetadas foram localizadas por pelo menos um dos quatorze sujeitos.

Desta forma, concluímos que os resultados obtidos com os quase-experimentos podem indicar a eficácia e aplicabilidade de nossa adaptação da técnica PBR para inspecionar documentos de especificação de requisitos de Sistemas Multiagentes. Além disso, os quase-experimentos contribuíram para aumentar a experiência em inspeções de software dos sujeitos participantes.

Nossa proposta de inspeção foi desenvolvida para inspecionar DERSs produzidos por um processo específico de engenharia de requisitos. Esses documentos são estruturados de acordo com o padrão ISO/IEC/IEEE 29148:2018 estendido para o contexto de sistemas multiagentes. Além disso, os diagramas de casos de uso e sua documentação

foram produzidos por meio da notação fornecida pela MASRML (GUEDES et al., 2020). No entanto, acreditamos que nossa técnica de inspeção pode ser adaptada para inspecionar outros DERSs produzidos por outros processos de engenharia de requisitos para sistemas multiagentes.

Esta pesquisa de mestrado é parte integrante de um trabalho maior que visa propor um processo completo de engenharia de requisitos para o desenvolvimento de Sistemas Multiagentes, ou seja, um processo que suporte as subáreas de Elicitação, Análise, Especificação e Validação de requisitos. Assim sendo, este estudo contribui com a proposição de novas abordagens para Especificação e Verificação de requisitos para Sistemas Multiagentes com ênfase no modelo BDI. As subáreas de Elicitação e Análise estão sendo desenvolvidas por outros pesquisadores em paralelo a este estudo.

Listamos abaixo alguns trabalhos que foram produzidos durante a execução desta pesquisa:

Publicados:

- Mendonça, Giovane D'Avila; Guedes, Gilleanes T. A. Metodologia Tropos: Uma Análise do Estado-da-Arte por Meio de uma Revisão Sistemática. In: ESCOLA REGIONAL DE ENGENHARIA DE SOFTWARE (ERES), 3., 2019, Rio do Sul. Anais [...]. Porto Alegre: Sociedade Brasileira de Computação, 2019 . p. 167-174.

- Guedes, G.; Souza Filho, I. P.; Gaerdicke, L. F.; Mendonça, G. D.; Vicari, R. M.; Brusius, C. MASRML - A Domain-Specific Modeling Language for Multi-Agent Systems Requirements. International Journal of Software Engineering & Applications(IJSEA), v. 11, n. 5, 2020. ISSN 0975-9018.

- Mendonça, G.; Filho, I. and Guedes, G. (2021). A Systematic Review about Requirements Engineering Processes for Multi-Agent Systems. In Proceedings of the 13th International Conference on Agents and Artificial Intelligence - Volume 1: ICAART, ISBN 978-989-758-484-8; ISSN 2184-433X, pages 69-79.

Aceito:

- Requirements Engineering Processes for Multi-Agent Systems - Chapter. Submetido ao HANDBOOK ON ARTIFICIAL INTELLIGENCE-ENHANCED SOFTWARE ENGINEERING a ser publicado pela Springer.

Convite artigo estendido:

- Requirements engineering processes for multi-agent systems through a systematic review. Será submetido como parte de um Livro da LNAI Series a ser publicado pela Springer.

Sob avaliação:

- Extending a Requirements Specification Standard for the Multiagent Systems Requirements Representation. Submetido ao *14th Conference on Artificial General Intelligence* (AGI-21) como Artigo regular.
- Validating Requirements Specification Documents for Multiagent Systems. Submetido ao *19th International Conference on Software Engineering and Formal Methods* (SEFM 2021) como Artigo Regular.

Futura submissão:

- A Requirements Engineering Process for BDI Model-Based Multiagent Systems. Futura submissão para um jornal.
- Será submetida a solicitação de depósito de patente do processo para desenvolvimento de sistemas multiagentes que inclui a técnica de inspeção e o padrão ISO/IEC/IEEE 29148:2018 estendido propostos neste estudo.

Como trabalhos futuros, pretende-se continuar estendendo o processo para desenvolvimento de sistemas multiagentes. Pretende-se ainda adicionar a fase de projeto ao processo atualmente em desenvolvimento. Pesquisaremos a viabilidade de desenvolver uma ferramenta para automatizar a especificação dos requisitos com base em nossa proposta de extensão do modelo do padrão ISO/IEC/IEEE 29148:2018. Também pesquisaremos a possibilidade de automatizar o processo de verificação de requisitos seguindo o GIRSMA. E por fim, executaremos outro quase-experimento ou experimento para avaliarmos a eficácia, eficiência e aplicabilidade de nossa técnica de inspeção envolvendo pesquisadores especialistas em Sistemas Multiagentes.

REFERÊNCIAS

ABUSHARK, Y. et al. Requirements specification via activity diagrams for agent-based systems. **Autonomous Agents and Multi-Agent Systems**, 02 2016. Citado 5 vezes nas páginas 44, 47, 49, 52 e 89.

ACKERMANN, C. et al. Assessing the quality impact of design inspections. In: **First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007)**. [S.l.: s.n.], 2007. p. 470–472. Citado 2 vezes nas páginas 102 e 108.

ALONSO, F. et al. Sonia: A methodology for natural agent development. In: . [S.l.: s.n.], 2004. p. 245–260. Citado 4 vezes nas páginas 42, 44, 47 e 49.

ALSANAD, A.; CHIKH, A. Reengineering of software requirement specification. In: _____. **New Perspectives in Information Systems and Technologies**. Springer, Cham: Springer, 2014. p. 95–111. ISBN 978-3-319-05948-. Disponível em: <https://doi.org/10.1007/978-3-319-05948-8_10>. Citado na página 89.

ANU, V. et al. Developing and evaluating learning materials to introduce human error concepts in software engineering courses: Results from industry and academia. In: **2019 IEEE Frontiers in Education Conference (FIE)**. [S.l.: s.n.], 2019. p. 1–9. Citado 3 vezes nas páginas 77, 82 e 104.

ARGENTE, E.; BOTTI, V.; JULIÁN, V. Gormas: An organizational-oriented methodological guideline for open mas. In: . [S.l.: s.n.], 2009. p. 32–47. Citado 4 vezes nas páginas 44, 47, 49 e 53.

ASHAMALLA, A.; BEYDOUN, G.; LOW, G. Model driven approach for real-time requirement analysis of multi-agent systems. **Computer Languages, Systems & Structures**, v. 50, 06 2017. Citado 3 vezes nas páginas 44, 47 e 49.

BANACH, R. A deidealisation semantics for kaos. In: . [S.l.: s.n.], 2010. p. 267–274. Citado 4 vezes nas páginas 44, 47, 49 e 64.

BASILI, V. R. et al. The empirical investigation of perspective-based reading. **Empirical Software Engineering**, v. 1, p. 133–164, 1996. Citado 4 vezes nas páginas 26, 28, 36 e 117.

BAUER, B. Extending uml for the specification of interaction protocols. **6th call for Proposal of FIPA**, 1999. Citado na página 74.

BERENBACH, B. et al. **Software & systems requirements engineering: in practice**. [S.l.]: McGraw-Hill, Inc., 2009. Citado na página 31.

BILAL, H. A. et al. Requirements validation techniques: An empirical study. **International Journal of Computer Applications**, Foundation of Computer Science (FCS), NY, USA, New York, USA, v. 148, n. 14, p. 5–10, Aug 2016. ISSN 0975-8887. Disponível em: <http://www.ijcaonline.org/archives/volume148/number14/25838-2016910911>. Citado 3 vezes nas páginas 26, 27 e 77.

BIOLCHINI, J. et al. **Systematic review in software engineering.** [S.l.]: System Engineering and Computer Science Department COPPE/UFRJ, 2005. Citado na página 39.

BJARNASON, E. et al. Challenges and practices in aligning requirements with verification and validation: a case study of six companies. **Empirical Software Engineering**, 2014. Citado na página 32.

BLANES, D.; INSFRAN, E.; ABRAHãO, S. Re4gaia: A requirements modeling approach for the development of multi-agent systems. In: . [S.l.: s.n.], 2009. v. 59, p. 245–252. Citado 3 vezes nas páginas 44, 47 e 49.

BOKMA, A. et al. Engineering large-scale agent-based systems with consensus. **Robotics and Computer-Integrated Manufacturing**, v. 11, n. 2, p. 81 – 90, 1994. ISSN 0736-5845. Disponível em: <http://www.sciencedirect.com/science/article/pii/0736584594900124>. Citado 4 vezes nas páginas 42, 44, 47 e 49.

BONJEAN, N. et al. Adelfe 2.0. In: _____. **Handbook on Agent-Oriented Design Processes**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014. p. 19–63. ISBN 978-3-642-39975-6. Disponível em: <https://doi.org/10.1007/978-3-642-39975-6_3>. Citado 8 vezes nas páginas 44, 45, 46, 47, 49, 68, 69 e 72.

BOUFEDJI, D. et al. Towards a mas product line engineering approach. In: _____. **Engineering Multi-Agent Systems - EMAS 2017**. Cham: Springer, 2018. p. 161–179. ISBN 978-3-319-91898-3. Disponível em: <https://doi.org/10.1007/978-3-319-91899-0_10>. Citado na página 90.

BOURQUE, P.; FAIRLEY, R. E. et al. **Guide to the software engineering body of knowledge (SWEBOK (R)): Version 3.0**. [S.l.]: IEEE Computer Society Press, 2014. Citado 4 vezes nas páginas 31, 32, 40 e 46.

BOYARCHUK, A. et al. Approach to the analysis of software requirements specification on its structure correctness. **CEUR Workshop Proceedings**, v. 2623, p. 85–95, 2020. ISSN 16130073. Disponível em: <http://ceur-ws.org/Vol-2623/paper9.pdf>. Citado 2 vezes nas páginas 82 e 89.

BRATMAN, M. et al. **Intention, plans, and practical reason**. [S.l.]: Harvard University Press Cambridge, MA, 1987. v. 10. 1-57586-192-5. Citado 2 vezes nas páginas 26 e 34.

BRATTHALL, L.; WOHLIN, C. Is it possible to decorate graphical software design and architecture models with qualitative information?-an experiment. **IEEE Transactions on Software Engineering**, v. 28, n. 12, p. 1181–1193, 2002. Citado na página 52.

BRESCIANI, P.; DONZELLI, P. Ref: A practical agent-based requirement engineering framework. In: . [S.l.: s.n.], 2003. v. 2814, p. 217–228. Citado 4 vezes nas páginas 44, 47, 49 e 58.

BRESCIANI, P. et al. Tropos: An agent-oriented software development methodology. **Autonomous Agents and Multi-Agent Systems**, p. 203–236, 2004. Citado na página 58.

BRYL, V. et al. B-tropos: Agent-oriented requirements engineering meets computational logic for declarative business process modeling and verification. In: . [S.l.: s.n.], 2008. p. 157–176. ISBN 9783540888321. Citado 4 vezes nas páginas 44, 47, 49 e 55.

CAIRE, G. et al. Agent oriented analysis using message/uml. In: . [S.l.: s.n.], 2001. p. 119–135. Citado 5 vezes nas páginas 44, 45, 47, 49 e 59.

CAO, L. Osoad methodology. In: _____. [S.l.: s.n.], 2015. p. 111–129. Citado 4 vezes nas páginas 44, 45, 47 e 49.

CAO, L. et al. Integrative early requirements analysis for agent-based systems. In: **Fourth International Conference on Hybrid Intelligent Systems (HIS'04)**. [S.l.: s.n.], 2004. p. 118–123. Citado 4 vezes nas páginas 44, 47, 49 e 63.

CARNEIRO, G. et al. Investigating the influence of inspector learning styles on design inspections: Findings of a quasi-experiment. **XX Ibero-American Conference on Software Engineering**, 2017. Citado na página 34.

CARVER, J.; NAGAPPAN, N.; PAGE, A. The impact of educational background on the effectiveness of requirements inspections: An empirical study. **IEEE Transactions on Software Engineering**, v. 34, p. 800–812, 2008. Citado na página 116.

CHAIPUNYATHAT, A. et al. A conceptual model of requirement engineering in cloud project delivery for thai government organizations. In: . [S.l.: s.n.], 2019. Cited By 0. Citado na página 25.

CHEBANYUK, O.; PALAHIN, O.; MARKOV, K. Domain engineering approach of software requirements analysis. p. 164–172, 2020. Citado na página 32.

CHOREN, R.; LUCENA, C. The anote modeling language for agent-oriented specification. In: _____. **Software Engineering for Multi-Agent Systems III: Research Issues and Practical Applications**. Berlin, Heidelberg: Springer-Verlag, 2005. p. 198–212. ISBN 3540248439. Citado 2 vezes nas páginas 35 e 94.

CIOLKOWSKI, M. et al. **Empirical Investigation of Perspective-based Reading: A Replicated Experiment**. 1997. Citado na página 117.

COSSENTINO, M. et al. A glimpse of the aspecs process documented with the fipa dpdf template. In: . [S.l.: s.n.], 2010. v. 627. Citado 4 vezes nas páginas 44, 47, 49 e 56.

COSSENTINO, M. et al. The aspecs process. In: _____. **Handbook on Agent-Oriented Design Processes**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014. p. 65–114. ISBN 978-3-642-39975-6. Disponível em: <https://doi.org/10.1007/978-3-642-39975-6_4>. Citado na página 117.

COSSENTINO, M. et al. The gaia methodology process. In: _____. **Handbook on Agent-Oriented Design Processes**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014. p. 141–172. ISBN 978-3-642-39975-6. Disponível em: <https://doi.org/10.1007/978-3-642-39975-6_6>. Citado 6 vezes nas páginas 44, 45, 47, 49, 61 e 72.

COSSENTINO, M. et al. **Handbook on Agent-Oriented Design Processes**. [S.l.]: Springer-Verlag Berlin Heidelberg, 2014. Citado 2 vezes nas páginas 41 e 70.

COSSENTINO, M.; SEIDITA, V. Passi: Process for agent societies specification and implementation. In: _____. [S.l.: s.n.], 2014. p. 287–329. Citado 5 vezes nas páginas 44, 45, 47, 49 e 71.

CUESTA, P.; GÓMEZ, A.; GONZÁLEZ, J. C. Agent oriented software engineering. Birkhäuser Verlag Base, 2007. Citado na página 25.

CYSNEIROS, L.; YU, E. Requirements engineering for large-scale multi-agent systems. In: . [S.l.: s.n.], 2002. v. 2603, p. 39–56. Citado 6 vezes nas páginas 44, 47, 49, 65, 68 e 69.

DELOACH, S.; GARCIA-OJEDA, J. The o-mase methodology. In: _____. [S.l.: s.n.], 2014. p. 253–285. Citado 7 vezes nas páginas 35, 44, 45, 47, 49, 60 e 90.

DELOACH, S.; WOOD, M. Developing multiagent systems with agenttool. In: . [S.l.: s.n.], 2000. v. 1986, p. 46–60. Citado na página 25.

DOMANN, J. et al. An agile method for multiagent software engineering. **Procedia Computer Science**, v. 32, p. 928 – 934, 2014. ISSN 1877-0509. The 5th International Conference on Ambient Systems, Networks and Technologies (ANT-2014), the 4th International Conference on Sustainable Energy Information Technology (SEIT-2014). Disponível em: <http://www.sciencedirect.com/science/article/pii/S1877050914007133>. Citado 3 vezes nas páginas 44, 47 e 49.

DUNSMORE, A.; ROPER, M.; WOOD, M. The development and evaluation of three diverse techniques for object-oriented code inspection. **IEEE Trans. Softw. Eng.**, IEEE Press, v. 29, n. 8, p. 677–686, ago. 2003. ISSN 0098-5589. Disponível em: <https://doi.org/10.1109/TSE.2003.1223643>. Citado na página 77.

DZIDA, W.; FREITAG, R. Making use of scenarios for validating analysis and design. **IEEE Transactions on Software Engineering**, v. 24, n. 12, p. 1182–1196, 1998. Citado na página 32.

EBAD, S. A. Inspection reading techniques applied to software artifacts - a systematic review. **Comput. Syst. Sci. Eng.**, v. 32, 2017. Citado 5 vezes nas páginas 33, 34, 36, 77 e 78.

ENGHOLM JÚNIOR, H. **Engenharia de Software na Prática**. [S.l.]: Novatec, 2010. ISBN 978-8575222171. Citado na página 32.

FAGAN, M. E. Design and code inspections to reduce errors in program development. **IBM Systems Journal**, v. 15, n. 3, p. 182–211, 1976. Citado na página 77.

FOGELSTRÖM, N.; GORSCHEK, T. Test-case driven versus checklist-based inspections of software requirements - an experimental evaluation. In: **Proceedings of the 10th Workshop on Requirements Engineering**. [S.l.: s.n.], 2007. p. 116–126. Citado 2 vezes nas páginas 26 e 77.

FONTELLES, M. J. et al. Metodologia da pesquisa científica: diretrizes para a elaboração de um protocolo de pesquisa. **Revista paraense de medicina**, v. 23, n. 3, p. 1–8, 2009. Citado na página 29.

FRANCH, X. et al. A study about the knowledge and use of requirements engineering standards in industry. **IEEE Transactions on Software Engineering**, 2021. Cited By 0. Citado 3 vezes nas páginas 26, 89 e 104.

FUENTES-FERNÁNDEZ, R.; GÓMEZ-SANZ, J.; PAVÓN, J. Requirements elicitation and analysis of multiagent systems using activity theory. **Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on**, v. 39, p. 282 – 298, 04 2009. Citado 4 vezes nas páginas 25, 44, 47 e 49.

FUENTES-FERNÁNDEZ, R.; GÓMEZ-SANZ, J. J.; PAVÓN, J. Verification and validation techniques for multi-agent systems. **Upgrade**, v. 5, p. 15–19, 2004. Citado 2 vezes nas páginas 25 e 27.

GALAFASSI, F. et al. Heráclito: Intelligent tutoring system for logic. In: _____. **Applications of Survivable Agents and Multi-Agent Systems: The PAAMS Collection**. Cham: Springer, 2019. p. 251–254. ISBN 978-3-030-24209-1. Disponível em: <https://doi.org/10.1007/978-3-030-24209-1_24>. Citado na página 118.

GALAFASSI, F. F. P. et al. Heráclito: Learning environment to teach logic. In: SPRINGER. **International Conference on Practical Applications of Agents and Multi-Agent Systems**. [S.l.], 2017. p. 316–320. Citado na página 100.

GANGADHARAN, A. **An Evaluation of Automatic Test Case Generation strategy from Requirements for Electric/Autonomous Vehicles**. 2020. Disponível em: <https://www.diva-portal.org/smash/get/diva2:1467129/FULLTEXT01.pdf>. Citado na página 97.

GAUR, V.; SONI, A. A novel approach to explore inter agent dependencies from user requirements. **Procedia Technology**, v. 1, p. 412–419, 12 2012. Citado 3 vezes nas páginas 44, 47 e 49.

GAZERANI, N. et al. Developing a teaching framework to support software inspection. In: **2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI)**. [S.l.: s.n.], 2015. p. 84–90. Citado na página 116.

GENZA, N.; MIGHELE, E. Review on multi-agent oriented software engineering implementation. **International Journal of Computer and Information Technology**, v. 2, May 2013. ISSN 2279–0764. Disponível em: <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.404.899&rep=rep1&type=pdf>. Citado na página 25.

GLASER, N. The comomas methodology and environment for multi-agent system development. **Multi-Agent Systems Methodologies and Applications**, v. 1286, p. 1–16, 1997. Australian Workshop on Distributed Artificial Intelligence. Disponível em: <https://link.springer.com/chapter/10.1007/BFb0030078>. Citado 4 vezes nas páginas 44, 45, 47 e 49.

GLINZ, M. **A Glossary of Requirements Engineering Terminology**. [S.l.]: International Requirements Engineering Board IREB, 2013. v. 1.5. Citado na página 89.

GONZÁLEZ-MORENO, J. et al. Ingenias-scrum. In: _____. [S.l.: s.n.], 2014. p. 219–251. Citado 5 vezes nas páginas 42, 44, 45, 47 e 49.

GUEDES, G. et al. Masrml - a domain-specific modeling language for multi-agent systems requirements. **International Journal of Software Engineering & Applications (IJSEA)**, v. 11, n. 5, 2020. ISSN 0975-9018. Citado 5 vezes nas páginas 26, 28, 71, 117 e 119.

GUEDES, G. T. A.; VICARI, R. M. A uml profile oriented to the requirements modeling in intelligent tutoring systems projects. In: BRAMER, M. (Ed.). **Artificial Intelligence in Theory and Practice III**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. p. 133–142. ISBN 978-3-642-15286-3. Citado na página 25.

GUPTA, A.; DERAMAN, A.; SIDDIQUI, S. Bdi logics for bdi architectures: old problems, new perspectives. **ARPN Journal of Engineering and Applied Sciences**, Asian Research Publishing Network, v. 14, n. 17, p. 3046–3061, 2019. Citado 2 vezes nas páginas 33 e 117.

HAJER, B. M.; TAIEB, B. R.; RAOUF, K. A new mas based approach modeling the qms continual improvement. In: **2009 IEEE International Conference on Systems, Man and Cybernetics**. [S.l.: s.n.], 2009. p. 4734–4739. Citado 4 vezes nas páginas 42, 44, 47 e 49.

HAUMER, P. et al. Bridging the gap between past and future in re: a scenario-based approach. In: **Proceedings IEEE International Symposium on Requirements Engineering (Cat. No.PR00188)**. [S.l.: s.n.], 1999. p. 66–73. Citado 5 vezes nas páginas 44, 47, 49, 63 e 68.

HEINZE, C.; PAPASIMEON, M.; GOSS, S. Specifying agent behaviour with use cases. In: **Proceedings of the Third Pacific Rim International Workshop on Multi-Agents: Design and Applications of Intelligent Agents**. Berlin, Heidelberg: Springer-Verlag, 2000. (PRIMA '00), p. 128–142. ISBN 3540679111. Citado 2 vezes nas páginas 71 e 74.

HERZIG, A. et al. Bdi logics for bdi architectures: old problems, new perspectives. **KI-Künstliche Intelligenz**, Springer, v. 31, n. 1, p. 73–83, 2017. Citado na página 35.

HILAIRE, V. et al. An approach for the integration of swarm intelligence in mas: An engineering perspective. **Expert Syst. Appl.**, Pergamon Press, Inc., USA, v. 40, n. 4, p. 1323–1332, mar. 2013. ISSN 0957-4174. Disponível em: <https://doi.org/10.1016/j.eswa.2012.08.058>. Citado 3 vezes nas páginas 44, 47 e 49.

HSIEH, M. et al. Spoken dialogue agent interface requirements modeling based on passi methodology. In: **2008 Eighth International Conference on Intelligent Systems Design and Applications**. [S.l.: s.n.], 2008. v. 1, p. 339–342. Citado 3 vezes nas páginas 44, 47 e 49.

HUIYING, X.; ZHI, J. An agent-oriented requirement graphic symbol representation and formalization modeling method. In: . [S.l.: s.n.], 2009. v. 4, p. 569 – 574. Citado 4 vezes nas páginas 44, 47, 49 e 54.

IEEE Recommended Practice for Software Requirements Specifications. **IEEE Std 830-1998**, p. 1–40, 1998. Citado 5 vezes nas páginas 71, 73, 88, 89 e 90.

IGLESIAS, C. et al. Analysis and design of multiagent systems using mas-commonkads. **Lecture Notes in Computer Science**, 11 1998. Citado 5 vezes nas páginas 44, 45, 47, 49 e 67.

IQBAL, D. et al. Requirement validation for embedded systems in automotive industry through modeling. **IEEE Access**, v. 8, p. 8697–8719, 2020. Citado na página 27.

ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes – Requirements engineering. **ISO/IEC/IEEE 29148:2018(E)**, p. 1–104, 2018. Citado 12 vezes nas páginas 26, 32, 33, 36, 37, 71, 82, 88, 89, 90, 96 e 117.

ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes –Requirements engineering. **ISO/IEC/IEEE 29148**, p. 1–94, 2011. Citado 2 vezes nas páginas 33 e 90.

JACKSON, M. **Problem Frames: Analyzing and Structuring Software Development Problems**. USA: Addison-Wesley Longman Publishing Co., Inc., 2000. ISBN 020159627X. Citado na página 52.

JAIN, S. A visualization technique for agent based goal refinement to elicit soft goals in goal oriented requirements engineering. In: . [S.l.: s.n.], 2007. p. 2–2. ISBN 978-0-7695-3248-6. Citado 3 vezes nas páginas 44, 47 e 49.

JENNINGS, N. R. Agent-oriented software engineering. In: GARIJO, F. J.; BOMAN, M. (Ed.). **Multi-Agent System Engineering**. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999. p. 1–7. ISBN 978-3-540-48437-0. Citado na página 25.

JO, C.-H.; EINHORN, J. A bdi agent-based software process. **Journal of Object Technology**, v. 4, p. 101–121, 11 2005. Citado 5 vezes nas páginas 42, 44, 47, 49 e 69.

JUAN, T.; PEARCE, A.; STERLING, L. Roadmap: Extending the gaia methodology for complex open systems. In: **Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems: Part 1**. New York, NY, USA: Association for Computing Machinery, 2002. (AAMAS '02), p. 3–10. ISBN 1581134800. Disponível em: <https://doi.org/10.1145/544741.544744>. Citado na página 56.

JURISTO, A. M. M. a. N. **Basics of Software Engineering Experimentation**. 1. ed. [S.l.]: Springer US, 2001. ISBN 978-1-4419-5011-6,978-1-4757-3304-4. Citado na página 99.

KHAN, H. et al. An empirical study of software requirements verification and validation techniques along their mitigation strategies. p. 2321–5658, 09 2015. Citado na página 33.

KITCHENHAM, B.; BRERETON, P. A systematic review of systematic review process research in software engineering. **Inf. Softw. Technol.**, Butterworth-Heinemann, USA, v. 55, n. 12, p. 2049–2075, dez. 2013. ISSN 0950-5849. Disponível em: <https://doi.org/10.1016/j.infsof.2013.07.010>. Citado na página 39.

KITCHENHAM, B.; CHARTERS, S. Guidelines for performing systematic literature reviews in software engineering. v. 2, 01 2007. Citado na página 39.

KOSCIANSKI, A.; SOARES, M. dos S. **Qualidade de Software-2ª Edição: Aprenda as metodologias e técnicas mais modernas para o desenvolvimento de software**. [S.l.]: Novatec Editora, 2007. Citado na página 25.

KRUCHTEN, P. **The Rational Unified Process – An Introduction**. [S.l.]: Addison Wesley Longman, 2000. Citado na página 59.

LABBA, C.; SAOUD, N. Bellamine ben; DUGDALE, J. Towards a conceptual framework to support adaptative agent-based systems partitioning. In: **2015 IEEE/ACIS 16th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)**. [S.l.: s.n.], 2015. p. 1–5. Citado na página 25.

LAITENBERGER, O. A survey of software inspection technologies. In: . [S.l.: s.n.], 2001. Citado na página 34.

LARSEN, J. B. Agent programming languages and logics in agent-based simulation. In: _____. **Modern Approaches for Intelligent Information and Database Systems**. Cham: Springer International Publishing, 2018. p. 517–526. ISBN 978-3-319-76081-0. Disponível em: <https://doi.org/10.1007/978-3-319-76081-0_44>. Citado na página 34.

LEE, C.-H.; LIU, A. A method for agent-based system requirements analysis. In: . [S.l.: s.n.], 2002. p. 214 – 221. ISBN 0-7695-1857-5. Citado 3 vezes nas páginas 44, 47 e 49.

LEE, J.; LEE, H. The sramo technique for analysis and reuse of requirements in multi-agent application engineering. **9th Workshop on Requirements Engineering**, p. 41–50, 2006. Citado 3 vezes nas páginas 44, 47 e 49.

LEE, J.; LEE, H. Strategic agent based web system development methodology. **International Journal of Information Technology & Decision Making**, v. 7, p. 309–337, 2008. Citado 3 vezes nas páginas 44, 47 e 49.

LEWIS, W. **Software Testing and Continuous Quality Improvement**. [S.l.]: CRC Press, 2009. Citado na página 34.

LIND, J. **Iterative Software Engineering for Multiagent Systems: The MASSIVE Method**. [S.l.: s.n.], 2001. ISBN 3-540-42166-1. Citado 4 vezes nas páginas 44, 45, 47 e 49.

LIU, L. et al. Agent-oriented requirements analysis from scenarios. In: O'SHEA, J. et al. (Ed.). **Agent and Multi-Agent Systems: Technologies and Applications**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. p. 394–405. ISBN 978-3-642-22000-5. Citado 3 vezes nas páginas 44, 47 e 49.

LIU, S. et al. Formal specification-based inspection for verification of programs. **IEEE Transactions on Software Engineering**, v. 38, n. 5, p. 1100–1122, 2012. Citado 2 vezes nas páginas 114 e 115.

LIU, W.; LI, M. Requirements planning with event calculus for runtime self-adaptive system. In: **2015 IEEE 39th Annual Computer Software and Applications Conference**. [S.l.: s.n.], 2015. v. 2, p. 77–82. Citado 3 vezes nas páginas 44, 47 e 49.

MALDONADO, J.; CARVER, J.; SHULL, F. Perspective-based reading: A replicated experiment focused on individual reviewer effectiveness. In: **Empirical Software Engineering**. [S.l.: s.n.], 2006. v. 11, p. 119–142. Citado 2 vezes nas páginas 114 e 115.

MENDONÇA, G.; SOUZA FILHO, I.; GUEDES, G. A systematic review about requirements engineering processes for multi-agent systems. In: **13th International Conference on Agents and Artificial Intelligence (ICAART 2021)**. [S.l.: s.n.], 2021. v. 1, p. 69–79. Citado 2 vezes nas páginas 25 e 39.

MORREALE, V. et al. Goal-oriented development of bdi agents: the practionist approach. In: . [S.l.: s.n.], 2006. p. 66–72. Citado 6 vezes nas páginas 35, 42, 44, 47, 49 e 66.

MUNROE, S. et al. Crossing the agent technology chasm: Lessons, experiences and challenges in commercial applications of agents. **Knowl. Eng. Rev.**, Cambridge University Press, USA, v. 21, n. 4, p. 345–392, dez. 2006. ISSN 0269-8889. Disponível em: <https://doi.org/10.1017/S0269888906001020>. Citado na página 89.

MURRAY, J. Specifying agent behaviors with uml statecharts and statedit. In: . [S.l.: s.n.], 2003. p. 145–156. Citado 3 vezes nas páginas 44, 47 e 49.

MYLOPOULOS, J.; CASTRO, J.; KOLP, M. The evolution of tropos. In: _____. [S.l.: s.n.], 2013. p. 281–287. ISBN 978-3-642-36925-4. Citado 7 vezes nas páginas 42, 44, 47, 49, 66, 69 e 72.

MüLLER, J.; FISCHER, K. Application impact of multi-agent systems and technologies: A survey. In: _____. **Agent-Oriented Software Engineering**. Springer, Berlin, Heidelberg: Springer, 2014. p. 27–53. ISBN 978-3-642-54432-3. Disponível em: <https://doi.org/10.1007/978-3-642-54432-3_3>. Citado na página 89.

NUSEIBEH, B.; EASTERBROOK, S. Requirements engineering: A roadmap. In: **Proceedings of the Conference on The Future of Software Engineering**. New York, NY, USA: Association for Computing Machinery, 2000. (ICSE '00), p. 35–46. ISBN 1581132530. Disponível em: <https://doi.org/10.1145/336512.336523>. Citado na página 31.

ODELL, J. J.; PARUNAK, H. V. D.; BAUER, B. Representing agent interaction protocols in uml. **Proceedings of the AAAI Agents Conference**, 2000. Citado na página 75.

PADGHAM, L.; WINIKOFF, M. Prometheus: A methodology for developing intelligent agents. In: . [S.l.: s.n.], 2002. Citado 5 vezes nas páginas 44, 45, 47, 49 e 52.

PADMANABAN, R. et al. Aose methodologies and comparison of object oriented and agent oriented software testing. In: . [S.l.: s.n.], 2016. p. 1–16. Citado na página 78.

PAGLIUSO, P.; TAMBASCIA, C.; VILLAS-BOAS, A. Melhoria da inspeção de requisitos segundo a técnica de leitura baseada em perspectiva. **XI Seminário de Computação - XI SEMINCO**, v. 32, 2002. Citado 3 vezes nas páginas 36, 79 e 80.

PASSOS, L. S.; ROSSETTI, R. J.; GABRIEL, J. Chapter 3 - an agent methodology for processes, the environment, and services. In: ROSSETTI, R. J.; LIU, R. (Ed.). **Advances in Artificial Transportation Systems and Simulation**. Boston: Academic Press, 2015. p. 37 – 53. ISBN 978-0-12-397041-1. Disponível em: <http://www.sciencedirect.com/science/article/pii/B9780123970411000030>. Citado 3 vezes nas páginas 44, 47 e 49.

PMI (Ed.). **A Guide to the Project Management Body of Knowledge (PMBOK Guide)**. 5. ed. Newtown Square, PA: Project Management Institute, 2013. ISBN 978-1-935589-67-9. Citado na página 33.

PORNPHOL, P.; CHITTAYASOTHORN, S. An agent-based quality assurance assessment system. **5th WSEAS International Conference on E-ACTIVITIES**, 2006. Citado na página 25.

PORTER, A.; VOTTA, L. An experiment to assess different defect detection methods for software requirements inspections. In: **Proceedings of 16th International Conference on Software Engineering**. [S.l.: s.n.], 1994. p. 103–112. Citado na página 77.

PRESSMAN, R. **Engenharia de Software: Uma Abordagem Profissional**. [S.l.]: Bookman, 2011. Citado na página 33.

RAJA, U. A. Empirical studies of requirements validation techniques. **2nd International Conference on ComputerBlekinge Institute of Technology**, 2009. Citado na página 33.

RANJAN, P.; MISRA, A. A novel approach of requirements analysis for agent based system. In: . [S.l.: s.n.], 2006. p. 299– 304. ISBN 0-7695-2611-X. Citado 5 vezes nas páginas 44, 47, 49, 55 e 56.

RANJAN, P.; MISRA, A. K. An enhanced model for agent based requirement gathering and pre-system analysis. In: **Proceedings of the 13th Annual IEEE International Symposium and Workshop on Engineering of Computer Based Systems**. USA: IEEE Computer Society, 2006. (ECBS '06), p. 187–195. ISBN 0769525466. Disponível em: <https://doi.org/10.1109/ECBS.2006.24>. Citado na página 56.

RAO, A. S.; GEORGEFF, M. P. Bdi agents: From theory to practice. In: **IN PROCEEDINGS OF THE FIRST INTERNATIONAL CONFERENCE ON MULTI-AGENT SYSTEMS (ICMAS-95**. [S.l.: s.n.], 1995. p. 312–319. Citado na página 34.

RIBINO, P. et al. Ontology and goal model in designing bdi multi-agent systems. In: . [S.l.: s.n.], 2013. v. 1099. Citado 5 vezes nas páginas 44, 47, 49, 53 e 69.

RODRIGUEZ, L. et al. Improving the quality of agent-based systems: Integration of requirements modeling into gaia. In: . [S.l.: s.n.], 2009. p. 278–283. Citado 3 vezes nas páginas 44, 47 e 49.

RONALD, N. et al. On the engineering of agent-based simulations of social activities with social networks. **Information and Software Technology**, v. 54, n. 6, p. 625 – 638, 2012. ISSN 0950-5849. Special Section: Engineering Complex Software Systems through Multi-Agent Systems and Simulation. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0950584911002485>. Citado 4 vezes nas páginas 44, 47, 49 e 62.

ROSA, J. C. S. et al. **Experimentando o SPIDe aplicado à Elicitação de Requisitos**. 2018. Disponível em: <http://wer.inf.puc-rio.br/WERpapers/artigos/artigos_WER18/WER_2018_paper_34.pdf>. Citado na página 31.

RUSSELL, S. J.; NORVIG, P. **Artificial intelligence: a modern approach**. [S.l.]: Malaysia; Pearson Education Limited,, 2016. Citado na página 35.

RYAN, M.; WHEATCRAFT, L. On the use of the terms verification and validation. **INCOSE International Symposium**, v. 27, p. 1277–1290, 07 2017. Citado 3 vezes nas páginas 26, 32 e 33.

SAAVEDRA, R.; BALLEJOS, L. C.; ALE, M. Quality properties evaluation for software requirements specifications: An exploratory analysis. In: **CIbSE**. [S.l.: s.n.], 2013. p. 245–258. Citado 2 vezes nas páginas 26 e 77.

SAITO, S. et al. Requirements clinic: Third party inspection methodology and practice for improving the quality of software requirements specifications. In: **2013 21st IEEE International Requirements Engineering Conference (RE)**. [S.l.: s.n.], 2013. p. 290–295. Citado na página 82.

SAQI, S. B.; AHMED, S. Requirements validation techniques practiced in industry: Studies of six companies. **Blekinge Institute of Technology**, 2008. Citado na página 33.

SAUER, C. et al. The effectiveness of software development technical reviews: a behaviorally motivated program of research. **IEEE Transactions on Software Engineering**, v. 26, n. 1, p. 1–14, 2000. Citado na página 116.

SCHNEIDER, G. M.; MARTIN, J.; TSAI, W. An experimental study of fault detection in user requirements documents. In: **ACM Transactions on Software Engineering and Methodology**. [S.l.: s.n.], 1992. v. 1, n. 2. Citado 2 vezes nas páginas 114 e 115.

SCHREIBER, G. et al. Commonkads: a comprehensive methodology for kbs development. **IEEE Expert**, v. 9, n. 6, p. 28–37, 1994. Citado na página 67.

SEN, A.; HEMACHANDRAN, K. Elicitation of goals in requirements engineering using agile methods. In: . [S.l.: s.n.], 2010. p. 263–268. Citado 3 vezes nas páginas 44, 47 e 49.

SEN, A. M.; JAIN, S. K. An agile technique for agent based goal refinement to elicit soft goals in goal oriented requirements engineering. In: **15th International Conference on Advanced Computing and Communications (ADCOM 2007)**. [S.l.: s.n.], 2007. p. 41–47. Citado 3 vezes nas páginas 44, 47 e 49.

SHAN, Q. et al. An empirical evaluation of capture-recapture estimators in software inspection. In: **2015 24th Australasian Software Engineering Conference**. [S.l.: s.n.], 2015. p. 58–67. Citado 3 vezes nas páginas 114, 115 e 118.

SHEN, Z. et al. Goal-oriented methodology for agent system development. In: . [S.l.: s.n.], 2005. E89-D, p. 95–101. Citado 3 vezes nas páginas 44, 47 e 49.

SHULL, F.; RUS, I.; BASILI, V. How perspective-based reading can improve requirements inspections. **Computer**, v. 33, n. 7, p. 73–79, 2000. Citado 6 vezes nas páginas 26, 28, 36, 78, 79 e 117.

SIVAKUMAR, N.; VIVEKAN, K.; KANIMOZHI, M. **Testing in Prometheus Methodology – Plan Oriented Approach**. 2013. Citado na página 25.

SLHOUB, K.; CARVALHO, M.; BOND, W. Recommended practices for the specification of multi-agent systems requirements. In: **2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON)**. [S.l.: s.n.], 2017. p. 179–185. Citado 8 vezes nas páginas 32, 71, 73, 88, 91, 92, 93 e 117.

SOMMERVILLE, I. **Software Engineering (7th Edition)**. [S.l.]: Pearson Addison Wesley, 2004. ISBN 0321210263. Citado na página 89.

SOMMERVILLE, I. **Engenharia de Software**. [S.l.]: Addison Wesley, 2007. Citado na página 33.

SOMMERVILLE, I. **Engenharia de Software**. [S.l.]: Pearson Prentice Hall, 2011. ISBN 978-8579361081. Citado na página 32.

SPROCK, A. S.; GALLEGOS, J. C. P.; VICARI, R. M. Fuzzy method of evaluation of instructionals techniques based on learning styles: Fuzzyils-instruction. **IEEE Latin America Transactions**, v. 1, p. 1–8, 2017. Citado na página 89.

SUHAIB, M. Conflicts identification among stakeholders in goal oriented requirements engineering process. **International Journal of Innovative Technology and Exploring Engineering (IJITEE)**, v. 8, 2019. Citado na página 32.

SUTCLIFFE, A. Requirements engineering for complex collaborative systems. In: . [S.l.: s.n.], 2001. p. 110–117. ISBN 0-7695-1125-2. Citado 4 vezes nas páginas 44, 47, 49 e 58.

TANG, R. F.; HUANG, X. Y. The application of requirement engineering model in large software development process. In: **Manufacturing Systems and Industry Application**. [S.l.]: Trans Tech Publications Ltd, 2011. (Advanced Materials Research, v. 267), p. 193–198. Citado na página 97.

THELIN, T.; RUNESON, P.; WOHLIN, C. An experimental comparison of usage-based and checklist-based reading. **IEEE Trans. Softw. Eng.**, IEEE Press, v. 29, n. 8, p. 687–704, ago. 2003. ISSN 0098-5589. Disponível em: <https://doi.org/10.1109/TSE.2003.1223644>. Citado na página 77.

THELIN, T. et al. Evaluation of usage-based reading—conclusions after three experiments. Kluwer Academic Publishers, USA, v. 9, n. 1–2, p. 77–110, mar. 2004. ISSN 1382-3256. Disponível em: <https://doi.org/10.1023/B:EMSE.0000013515.86806.d4>. Citado na página 77.

UJJWAL, K.; CHODOROWSKI, J. A case study of adding proactivity in indoor social robots using belief–desire–intention (bdi) model. **Biomimetics**, v. 4, n. 74, December 2019. Citado na página 35.

ULFAT-BUNYADI, N.; MOHAMMADI, N.; HEISEL, M. Supporting the systematic goal refinement in kaos using the six-variable model. In: . [S.l.: s.n.], 2018. p. 102–111. Citado 4 vezes nas páginas 44, 47, 49 e 51.

UML 2.4.1 Superstructure Specification. 2021. Disponível em: <https://www.omg.org/spec/UML/2.5/About-UML/>. Citado 8 vezes nas páginas 53, 57, 59, 60, 64, 68, 71 e 74.

VICARI, R. M.; GLUZ, J. C. An intelligent tutoring system (its) view on aose. **International Journal of Agent-Oriented Software Engineering**, Inderscience Publishers, v. 1, n. 3-4, p. 295–333, 2007. Citado 2 vezes nas páginas 25 e 34.

VIE JR., D. L. **Writing Software Requirements Specifications**. 2021. Disponível em: <https://techwhirl.com/writing-software-requirements-specifications/>. Citado 2 vezes nas páginas 26 e 37.

VIEIRA, J. G. S. Metodologia de pesquisa científica na prática. **Curitiba: Editora Fael**, 2010. Citado na página 29.

WANG, W.; JIANG, Y. Multiagent-based allocation of complex tasks in social networks. **IEEE Transactions on Emerging Topics in Computing**, v. 3, n. 4, p. 571–584, 2015. Cited By 13. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84961675169&doi=10.1109%2fTETC.2015.2403200&partnerID=40&md5=f4be5d1921e14552754ab8ae2fcfb742>. Citado na página 25.

WANG, Y. et al. Plant: A pattern language for transforming scenarios into requirements models. **International Journal of Human-Computer Studies**, v. 71, p. 1026–1043, 11 2013. Citado 4 vezes nas páginas 44, 47, 49 e 62.

WIDODO, A. P. et al. Proposed technical activity on requirement engineering process. **Journal of Physics: Conference Series**, IOP Publishing, v. 1943, n. 1, p. 012103, jul 2021. Disponível em: <https://doi.org/10.1088/1742-6596/1943/1/012103>. Citado na página 27.

WILMANN, D.; STERLING, L. Guiding agent-oriented requirements elicitation: Homer. In: **Fifth International Conference on Quality Software (QSIC'05)**. [S.l.: s.n.], 2005. p. 419–424. Citado 3 vezes nas páginas 44, 47 e 49.

WOHLIN, C. **Experimentation in software engineering**. [S.l.]: Springer, 2012. ISBN 978-3-642-29043-5,978-3-642-29044-2,3642290434,3642290442. Citado 6 vezes nas páginas 29, 99, 102, 103, 104 e 105.

WOHLIN, C. et al. **Experimentation in Software Engineering: An Introduction**. 1. ed. [S.l.]: Springer US, 2000. (The Kluwer International Series in Software Engineering 6). ISBN 978-1-4613-7091-8,978-1-4615-4625-2. Citado na página 99.

WOOD, M. F.; DELOACH, S. A. An overview of the multiagent systems engineering methodology. In: **First International Workshop, AOSE 2000 on Agent-Oriented Software Engineering**. Berlin, Heidelberg: Springer-Verlag, 2001. p. 207–221. ISBN 3540415947. Citado na página 72.

WOOLDRIDGE, M. **An introduction to multiagent systems**. [S.l.]: John Wiley & Sons, 2009. Citado 2 vezes nas páginas 25 e 34.

WOOLDRIDGE, M.; JENNINGS, N. R. Intelligent agents: Theory and practice. **The knowledge engineering review**, Cambridge University Press, v. 10, n. 2, p. 115–152, 1995. Citado 2 vezes nas páginas 25 e 71.

WU, H.; LIU, L.; MA, W. Optimizing requirements elicitation with an i* and bayesian network integrated modelling approach. In: **2010 IEEE 34th Annual Computer Software and Applications Conference Workshops**. [S.l.: s.n.], 2010. p. 182–188. Citado 3 vezes nas páginas 44, 47 e 49.

YOUSUF, F.; ZAMAN, Z.; IKRAM, N. Requirements validation techniques in gsd: A survey. **IEEE International Multitopic Conference**, 2009. Citado na página 33.

YU, E. S.-K. **Modelling Strategic Relationships for Process Reengineering**. Tese (Doutorado) — University of Toronto, CAN, 1996. UMI Order No. GAXNN-02887 (Canadian dissertation). Citado 4 vezes nas páginas 35, 53, 58 e 65.

ZHI, S. M. Q. An evaluation of value-oriented review for software requirements specification. **Computer Systems Science and Engineering**, v. 37, n. 3, p. 443–461, 2021. Disponível em: <http://www.techscience.com/csse/v37n3/41713>. Citado na página 89.

Anexos

ANEXO A – DOCUMENTO DE ESPECIFICAÇÃO DE REQUISITOS PARCIAL DO SISTEMA HERÁCLITO

Documento de Especificação de Requisitos para Sistemas Multiagentes com base na ISO/IEC/IEEE 29148:2018 estendida.

Histórico da Revisão

Data <dd/mm/aaaa>	**Versão** <x.x>	**Descrição** <detalhes>	**Autor** <nome>
21/07/2021	1.0	Especificação Parcial do Sistema Heráclito	Giovane

1. Introdução

1.1. Propósito

O sistema será voltado ao aprendizado interativo e dialético com foco no ensino de dedução natural da lógica proposicional.

1.2. Escopo

1.2.1. Nome do produto de software

O sistema é denominado Heráclito.

1.2.2. O que o produto de software fará

O sistema permitirá que o aluno desenvolva o aprendizado de lógica. O Ambiente Heráclito visa auxiliar estudantes a resolver vários tipos de exercícios de Lógica, começando com exercícios para calcular o valor lógico de uma fórmula, passando por exercícios de construção de tabelas-verdade e chegando até exercícios de elaboração de provas de argumentos por meio das regras da Dedução Natural.

1.2.3. Aplicação do produto de software

O sistema tem como objetivo reduzir os altos índices de reprovação e desistência que ocorrem na disciplina de Lógica, tanto no ensino presencial quanto no ensino à distância. Com aprendizado interativo e dialético o aluno inicia a resolução de questões de lógica proposicional que são submetidas para avaliação do sistema. O sistema será desenvolvido utilizando o conceito de agentes. Estes agentes agirão no ambiente do sistema por meio de verificação de ações do aluno, aplicação de reforços de conteúdo, verificação de ociosidade do aluno, entre outras funcionalidades, que visam conduzir o aluno ao aprendizado de lógica proposicional.

1.3. Visão geral do produto

1.3.1. Perspectiva de Produto

O ator aluno interage com o sistema por meio de um editor de provas para resolução de exercícios de dedução de lógica proposicional. Ao resolver o exercício o aluno pode editá-lo, manipulando-o através do uso das regras, recebendo auxílio e ainda pode ou não solicitar ajuda do sistema tutor da ferramenta.

1.3.2. Perspectiva do Domínio

1.3.2.1. Ambiente Operacional

O sistema deve permitir que o aluno:

- Mantenha seu cadastro atualizado;
- Realize a prova;
- Resolva os exercícios;

O sistema deve:

- Verificar se o aluno consegue ou não aplicar as regras de dedução de forma correta;

1.3.2.2. Ambiente Físico/Virtual

O sistema Heráclito foi disponibilizado em versão desktop para ser rodado na máquina do aluno utilizando um arquivo de extensão .jar.

1.3.2.3. Domínio da Aplicação

O sistema é um ambiente virtual de aprendizagem com foco no ensino de dedução natural da lógica proposicional. Por meio deste sistema os alunos interagem para resolução de exercícios com interação de agentes tutores que auxiliarão na correção dos exercícios.

1.3.3. Perspectiva de Comunicação

1.3.3.1. Definir forma de comunicação entre agentes e usuários externos

Não definidos.

1.3.3.2. Tipo de informação a ser transmitida entre os agentes e usuários externos

Não definidos.

1.3.3.3. Forma de comunicação entre os agentes

Não definidos.

1.3.3.4. Padrão de comunicação dos agentes

Não definidos.

1.3.3.5. Tipo de informação a ser transmitida entre os agentes.

Não definidos.

1.3.4. Perspectiva do ambiente

Descrever características genéricas do ambiente.

Características do Ambiente	
Plataforma	Web
Arquitetura	BDI
Papéis de agente	Especialista Técnico, Mediador e Analista Bayesiano.

1.3.5. Funções do Produto

1.3.5.1. Funcionalidades que podem ser acessadas por usuários normais

Funcionalidade
Manter cadastro
Resolver questão
Realizar prova

1.3.5.2. Funcionalidades que são acessadas por papéis de agentes

Tipo	Funcionalidade	Descrição	Papel de agente associado
Objetivo	Verificar prova do aluno	Tem como propósito verificar se um aluno finalizou a prova.	Especialista Técnico
Objetivo	Verificar questão do aluno	Tem como propósito verificar a questão da prova finalizada	Especialista Técnico

		pelo aluno.	
Objetivo	Verificar ação do aluno	Tem como objetivo monitorar as ações do aluno dentro do sistema para verificar a finalização de uma prova.	Especialista Técnico
Objetivo	Planejar questão	O agente que assume o papel de especialista técnico executa este objetivo para planejar as questões destinadas ao aluno.	Especialista Técnico
Objetivo	Planejar sequência de conteúdo	Este objetivo é executado pelo agente que assume o papel de Especialista Técnico para planejar a sequência de conteúdo que melhor se adequa ao aluno.	Especialista Técnico
Objetivo	Informar questão ao aluno	Este objetivo é executado pelo agente que assume a função de Mediador para informar nova questão de estudo ao aluno.	Mediador
Objetivo	Definir estratégia de ensino	Este objetivo é executado pelo agente que assume a função de Mediador para definir uma estratégia de	Mediador

		ensino.	
Objetivo	Aplicar reforço ao aluno	Este objetivo é executado pelo agente que assume a função de Mediador para aplicar reforço ao aluno.	Mediador
Objetivo	Fornecer avaliação de questão	Este objetivo é executado pelo agente que assume a função de Mediador para retornar o resultado da questão ao aluno.	Mediador
Objetivo	Fornecer resultado de prova	Este objetivo é executado pelo agente que assume a função de Mediador para retornar o resultado ao aluno com relação a uma prova.	Mediador
Objetivo	Informar erro em questão do aluno	Este objetivo é executado pelo agente que assume a função de Mediador para informar o aluno caso ocorra um erro ou ação perigosa.	Mediador
Objetivo	Informar ociosidade do aluno	Este objetivo é executado pelo pelo agente que assume a função de Mediador para	Mediador

		enviar uma mensagem ao aluno em caso de ociosidade.	
Objetivo	Fornecer ajuda ao aluno	Este objetivo é executado pelo agente que assume a função de Mediador para retornar uma ajuda ao aluno.	Mediador
Objetivo	Informar perfil do aluno	Este objetivo é executado pelo agente que assume a função de Mediador a informar perfil do aluno.	Mediador
Objetivo	Informar rede bayesiana do aluno	Este objetivo é executado pelo agente que assume a função de “Analista Bayesiano” a enviar a rede Bayesiana.	Analista Bayesiano

1.3.6. Características dos usuários

Não definido.

1.3.7. Limitações

Requisito não-funcional	Descrição
Usabilidade	O editor de provas deve ser de fácil uso.

Interface	O editor de provas deve possuir um design adequado ao informar o conteúdo de textos e ao posicionar imagens/figuras.

1.4. Definições

Termo	Definição
Papéis de agentes	Representam as funções que os agentes podem desempenhar dentro de um sistema. Os agentes podem assumir mais de um papel, mas geralmente não ao mesmo tempo.
AgentRoleActors	Representam papéis assumidos por agentes que podem ter estereótipos do tipo ReactiveAgentRoles ou CognitiveAgentRoles, esses papéis de agentes são modelados dentro da fronteira do sistema.
Stereotype	Os estereótipos são um mecanismo de extensibilidade da UML. Eles dão mais poder à UML, permitindo classificar elementos com algo em comum.

2. Referências

Norma ISO/IEC/IEEE 29148:2018.

3. Requisitos

3.1. Funções

Não definido.

3.2. Requisitos de performance

Não se aplica.

3.3. Requisitos de usabilidade

Não se aplica.

3.4. Requisitos de interface

Não se aplica.

3.5. Requisitos lógicos de banco de dados
Não se aplica.

3.6. Restrições de projeto
Não se aplica.

3.7. Atributos de sistemas e software
Não definido.

3.8. Informações de apoio
Não definido.

3.9. Atribuição de papéis de agentes
Descrever as atribuições associadas ao papel de agente.

Papel de Agente	Descrição da Atribuição
Especialista Técnico	O Especialista Técnico é responsável por toda a parte que necessita conhecimento técnico. Este papel de agente prepara o conteúdo do aluno, corrige provas e questões e acompanha o aluno a cada ação, na resolução de uma questão.
Mediador	Único responsável pelo intermédio do sistema com o aluno. Acompanha o desenvolvimento do aluno, apresentando feedback e dando dicas. Este papel de agente também conhece o perfil do aluno.

3.10. Perspectiva de papéis de agentes
Esta etapa permite identificar as características do papel de agente.

3.10.1. Crenças iniciais do papel de agente
Descrever as crenças iniciais do papel de agente.

Papel de Agente	Crença
Especialista Técnico	Necessidade de nova verificação de prova negativa
	Aluno está executando a prova

	Solicitação de uma nova verificação de questão negativa
	Não houve solicitação de uma nova verificação de questão
	Necessidade de verificar questão em tempo de execução negativa
	O aluno não realizou uma ação
	Solicitação de uma nova questão negativa
	Não houve solicitação de uma nova questão para o aluno
	Perfil não enviado
	Solicitação de uma nova sequência de conteúdo negativa
	Não houve solicitação de uma nova sequência de conteúdo pelo mediador
Mediador	Solicitação de uma nova questão negativa
	Não houve solicitação de nova questão
	O Especialista Técnico não enviou uma nova questão
	Necessidade de definir estratégias de ensino negativa
	Não houve novo cadastro de aluno no sistema
Analista Bayesiano	Solicitação de rede Bayesiana do aluno negativa
	Não houve solicitação de envio de rede Bayesiana do aluno

3.10.2. Crenças necessárias para objetivo se tornar uma intenção

Descrever as crenças necessárias para o objetivo se tornar uma intenção

Objetivo	Crença necessária
Verificar prova do aluno	Necessidade de nova verificação de prova positiva
Verificar questão do aluno	Solicitação de uma nova verificação de questão positiva

Verificar ação do aluno	Necessidade de verificar questão em tempo de execução positiva
Planejar questão	Solicitação de uma nova questão positiva
Planejar sequência de conteúdo	Solicitação de uma nova sequência de conteúdo positiva
Informar questão ao aluno	Solicitação de uma nova questão positiva
Definir estratégia de ensino	Necessidade de definir estratégias de ensino positiva
Aplicar reforço ao aluno	Necessidade de aplicar reforço positiva
Fornecer avaliação de questão	Necessidade de fornecer resultado de questão positiva.
Fornecer resultado de prova	Necessidade de fornecer resultado positiva
Informar erro em questão ao aluno	Necessidade de fornecer retorno ao aluno positiva
Informar ociosidade do aluno	Necessidade de informar ociosidade positiva
Fornecer ajuda ao aluno	Necessidade de fornecer ajuda positiva
Informar perfil do aluno	Solicitação de perfil do aluno positiva
Informar rede bayesiana do aluno	Solicitação de rede Bayesiana positiva

3.10.3. Percepções associadas ao papel de agente

Descrever as percepções associadas ao papel de agente.

Papel de Agente	Percepção
Especialista Técnico	Aluno finalizou uma prova.

3.10.4. Crenças que podem ser afetadas pelas percepções

Descrever crenças que podem ser afetadas pelas percepções

Crença Associada	Percepção	Papel de Agente
Necessidade de nova verificação de prova negativa	Sondagem de novas finalizações de prova	Especialista Técnico
Solicitação de uma nova verificação de questão negativa	Sondagem de novas solicitações de de verificação de questões	Especialista Técnico
Necessidade de verificar questão em tempo de execução negativa	Sondagem de novas ações do aluno	Especialista Técnico
Necessidade de definir estratégias de ensino negativa	Sondagem de falha em prova	Mediador
Necessidade de fornecer resultado de questão negativa	Sondagem de término de questão	Mediador

3.10.5. **Possíveis ações externas associadas ao objetivo**

Descrever possíveis ações externas associadas ao objetivo

Ação	Objetivo Associado
Informar ao Mediador as informações referente ao resultado da prova	Verificar prova do aluno
Informar a avaliação da questão ao mediador	Verificar questão do aluno
Enviar uma mensagem de apoio ao aluno, avisando que o aluno está perto de terminar a prova	Fornecer resultado de prova
Enviar uma mensagem para verificar a razão da ociosidade do aluno	Informar ociosidade do aluno
Enviar perfil do aluno ao Especialista Técnico	Informar perfil do aluno

3.10.6. Possíveis planos associados ao objetivo

Descrever possíveis planos associados ao objetivo

Plano	Objetivo associado
Avaliar questão	Verificar prova de aluno
Avaliar questão	Verificar questão do aluno

3.10.7. Possíveis ações externas associadas ao plano

Descrever possíveis ações externas associadas ao plano

Ações	Plano Associado
Informar Mediador da anomalia encontrada na questão do aluno	Verificar questão em tempo de execução
Solicitar o perfil do aluno ao Mediador	Selecionar questão na base de conhecimento
Solicitar a rede bayesiana do aluno ao Analista Bayesiano	Selecionar questão na base de conhecimento
Informar sequência de conteúdo ao Mediador	Determinar conteúdo
Solicitar a rede bayesiana do aluno ao Analista Bayesiano	Determinar conteúdo
Solicitar o perfil do aluno ao Mediador	Determinar conteúdo
Aplicar questão ao aluno	Preparar aplicação da questão
Solicitar nova questão ao Especialista Técnico	Preparar aplicação da questão
Solicitar ao Analista Bayesiano o envio da rede bayesiana	Atualizar perfil do aluno
Solicitar sequência de conteúdo ao Especialista Técnico	Atualizar estratégia de ensino
Aplicar reforço com base no novo conteúdo	Preparar Reforço
Solicitar sequência de conteúdo ao Especialista Técnico	Preparar Reforço
Solicitar verificação da questão ao Especialista Técnico	Preparar avaliação de questão
Enviar mensagem ao aluno contendo o resultado da questão	Preparar avaliação de questão
Solicitar verificação da prova ao Especialista Técnico	Preparar verificação da prova
Enviar mensagem ao aluno contento feedback da prova	Preparar verificação da prova
Informar o aluno que a regra de dedução utilizada está incorreta	Preparar retorno de anomalia na questão

Informar o aluno de que a regra de dedução usada pode não estar correta para esta questão	Preparar retorno de anomalia na questão
Informar aluno que hipótese está incorreta	Preparar retorno de anomalia na questão
Enviar uma mensagem ao aluno contendo a ajuda	Planejar ajuda do aluno
Solicitar verificação da questão ao Especialista Técnico	Planejar ajuda do aluno
Enviar a rede bayesiana do aluno	Atualizar rede bayesiana do aluno

3.11. Perspectiva de casos de uso internos

3.11.1. Casos de uso interno do tipo Goal

Nome do caso de uso interno	Verificar prova do aluno
Stereotype	Goal
AgentRoleActor	Especialista Técnico
Resumo	Este caso de uso interno descreve as etapas seguidas pelo agente que assume a função de Especialista Técnico para verificar uma prova realizada pelo aluno.
Crenças Iniciais	Necessidade de nova verificação de prova negativa
Percepções	Sondagem de novas finalizações de prova.
Cenário principal	
Ações do AgentRoleActor	
1. Executar o caso de uso interno “Perceber que um aluno finalizou uma prova”	
Cenário Alternativo - Prova finalizada percebida	
Ações do AgentRoleActor	
1. Alterar crença para Necessidade de nova verificação de prova positiva	
2. Tornar o objetivo uma intenção	
3. Para cada uma das questões da prova execute o caso de uso interno “Avaliar questão”	
4. Inserir na prova a avaliação de cada questão.	
5. Realizar o cálculo de nota do aluno com base na correção realizada no plano.	
6. Informar ao Mediador o resultado da prova.	

Nome do caso de uso interno	Verificar questão do aluno
Stereotype	Goal
AgentRoleActor	Especialista Técnico
Resumo	Este caso de uso interno descreve as etapas seguidas pelo agente que assume a função de Especialista Técnico a verificar uma questão desenvolvida pelo aluno.
Crenças Iniciais	Solicitação de uma nova verificação de questão negativa
Percepções	Sondagem de novas solicitações de verificação de questões
Cenário principal	
Ações do AgentRoleActor	
1. Executar o caso de uso interno “Perceber solicitação de verificação de questão”	

Nome do caso de uso interno	Verificar ação do aluno
Stereotype	Goal
AgentRoleActor	Especialista Técnico
Resumo	Este caso de uso interno descreve as etapas seguidas pelo agente que assume a função de Especialista Técnico para verificar uma ação do aluno em tempo de execução.
Crenças Iniciais	Necessidade de verificar questão em tempo de execução negativa.
Percepções	Sondagem de novas ações do aluno.
Cenário Alternativo - Prova finalizada percebida	
Ações do AgentRoleActor	
1. Alterar crença para Necessidade de verificar questão em tempo de execução positiva.	
2. Tornar o objetivo uma intenção	
3. Executar caso de uso interno “Verificar questão em tempo de execução”	

Nome do caso de uso interno	Planejar sequência de conteúdo
Stereotype	Goal
AgentRoleActor	Especialista Técnico
Resumo	Este caso de uso interno descreve as etapas seguidas pelo agente que assume a função de Especialista Técnico para planejar a sequência de conteúdo que melhor se adequa ao aluno.
Crenças Iniciais	Solicitação de uma nova sequência de conteúdo negativa.
Percepções	Sondagem de novas solicitações de sequência de conteúdo.

Cenário principal
Ações do AgentRoleActor
1. Executar o caso de uso interno “Perceber solicitação de nova sequência de conteúdo”
Cenário Alternativo - Solicitação de Verificação Percebida
Ações do AgentRoleActor
1. Alterar crença para Solicitação de uma nova sequência de conteúdo positiva.
2. Tornar o objetivo uma intenção.
3. Executar caso de uso interno “Determinar conteúdo”

3.11.2. Casos de uso interno do tipo Perception

Nome do caso de uso interno	Perceber que um aluno finalizou uma prova
Stereotype	Perception
AgentRoleActor	Especialista Técnico
Resumo	Este caso de uso interno descreve a execução da sondagem do ambiente para verificar se um aluno finalizou uma prova.
Pré-condições	O caso de uso interno Goal “Verificar prova do Aluno”
Crenças Iniciais	Aluno está executando a prova.
Cenário principal	
Ações do AgentRoleActor	
1. Sondar o ambiente para verificar se um aluno finalizou uma prova.	
Cenário Alternativo - Percepção Verdadeira	
1. Passar a acreditar que o aluno finalizou a prova.	

Nome do caso de uso interno	Perceber solicitação de verificação de questão pelo Mediador
Stereotype	Plan
AgentRoleActor	Especialista Técnico
Resumo	Este caso de uso interno descreve a execução da sondagem do ambiente para verificar se uma nova verificação de questão foi solicitada.
Pré-condições	O caso de uso interno Goal “Verificar Questão do Aluno” deve estar em execução.

Crenças Iniciais	Não houve solicitação de uma nova verificação de questão
Cenário principal	
Ações do AgentRoleActor	
1. Sondar o ambiente para verificar se um aluno finalizou uma prova.	
Cenário Alternativo - Percepção Verdadeira	
1. Passar a acreditar que o aluno finalizou a prova.	

Nome do caso de uso interno	Perceber ação do aluno
Stereotype	Perception
AgentRoleActor	Especialista Técnico
Resumo	Este caso de uso interno descreve a execução da sondagem que o especialista Técnico faz na questão, buscando ações do aluno para verificar a questão em tempo de execução
Crenças Iniciais	O aluno não realizou uma ação.
Cenário principal	
Ações do AgentRoleActor	
1. Sondar o ambiente para verificar possíveis ações do aluno em uma questão.	
Cenário Alternativo - Percepção Verdadeira	
1. Passar a acreditar que o aluno realizou uma ação.	

Nome do caso de uso interno	Perceber solicitação de nova questão pelo Mediador
Stereotype	Perception
AgentRoleActor	Especialista Técnico
Resumo	Este caso de uso interno descreve a execução da sondagem do ambiente para verificar se uma nova questão foi solicitada
Pré-condições	O caso de uso interno Goal “Planejar questão” deve estar em execução
Crenças Iniciais	Não houve solicitação de um nova questão para o aluno
Cenário Alternativo - Percepção Verdadeira	

1. Passar a acreditar que uma nova questão foi solicitada.
2. Receber questão.

Nome do caso de uso interno	Perceber envio do perfil do aluno pelo Mediador
Stereotype	Perception
AgentRoleActor	Especialista Técnico
Resumo	Este caso de uso interno descreve a execução da sondagem do ambiente para verificar se o perfil do aluno foi enviado pelo Mediador.
Pré-condições	O caso de uso interno Plan “Selecionar questão na base de conhecimento” deve estar em execução.
Crenças Iniciais	Perfil não enviado
Cenário principal	
Ações do AgentRoleActor	
1. Sondar o ambiente para verificar a ocorrência de um novo envio de perfil do aluno por parte do Mediador.	
Cenário Alternativo - Percepção Verdadeira	
1. Passar a acreditar que um perfil de aluno foi enviado	
2. Receber o perfil do aluno.	

Nome do caso de uso interno	Perceber envio de rede bayesiana do aluno pelo Analista Bayesiano
Stereotype	Perception
AgentRoleActor	Especialista Técnico
Resumo	Este caso de uso interno descreve a execução da sondagem do ambiente para verificar se a rede bayesiana do aluno foi enviada pelo Analista Bayesiano.
Pré-condições	O caso de uso interno Plan “Selecionar questão na base de conhecimento” deve estar em execução.
Crenças Iniciais	Perfil não enviado

Cenário principal
Ações do AgentRoleActor
1. Sondar o ambiente para verificar a ocorrência de um novo envio de rede bayesiana do aluno por parte do Analista Bayesiano.
Cenário Alternativo - Percepção Verdadeira
1. Passar acreditar que uma rede bayesiana foi enviada
2. Receber rede bayesiana do aluno.

Nome do caso de uso interno	Perceber solicitação de nova sequência de conteúdo pelo Mediador
Stereotype	Perception
AgentRoleActor	Especialista Técnico
Resumo	Este caso de uso interno descreve a execução da sondagem do ambiente para verificar se uma nova sequência de conteúdo foi solicitada.
Pré-condições	O caso de uso interno Goal “Planejar sequência de conteúdo” deve estar em execução
Crenças Iniciais	Não houve solicitação de uma nova sequência de conteúdo pelo mediador
Cenário principal	
Ações do AgentRoleActor	
1. Sondar o ambiente para verificar a ocorrência de uma nova solicitação de sequência de conteúdo para um aluno.	
Cenário Alternativo - Percepção Verdadeira	
1. Passar a acreditar que uma nova sequência de conteúdo foi solicitada.	

3.11.3. Casos de uso interno do tipo Plan

Nome do caso de uso interno	Avaliar questão
Stereotype	Plan
AgentRoleActor	Especialista Técnico
Resumo	Este caso de uso interno descreve as etapas seguidas pelo agente que assume a função de Especialista Técnico consultar base de conhecimento para verificar a questão do aluno.
Cenário Principal	

Ações do AgentRoleActor
1. Verificar a base de conhecimento
2. Encontrar questão sob análise
3. Comparar a questão correta com a questão do aluno
4. Verificar questão do aluno procurando possíveis erros de dedução e erros de hipótese.
5. Descrever uma avaliação da questão com base na comparação e nos possíveis erros encontrados.

Nome do caso de uso interno	Verificar questão em tempo de execução
Stereotype	Plan
AgentRoleActor	Especialista Técnico
Resumo	Este caso de uso interno descreve as etapas seguidas pelo agente que assume a função de Especialista Técnico consultar base de conhecimento para verificar a questão do aluno em tempo de execução.
Cenário Principal	
Ações do AgentRoleActor	
1. Verificar a base de conhecimento	
2. Encontrar questão sob análise	
3. Comparar a questão correta com a questão do aluno	
4. Identificar que a questão do aluno está correta e não possui ações perigosas.	
Cenário Alternativo - Regra de dedução incorreta.	
Ações do AgentRoleActor	
1. Identificar que regra de dedução utilizada pelo aluno está incorreta.	
2. Descrever aplicação de regra de dedução incorreta na questão do aluno.	
3. Informar Mediador da anomalia encontrada na questão do aluno.	
Cenário Alternativo - Hipótese incorreta.	
Ações do AgentRoleActor	
1. Identificar que hipótese utilizada pelo aluno está incorreta.	
2. Descrever aplicação de Hipótese incorreta na questão do aluno.	
3. Informar Mediador da anomalia encontrada na questão do aluno.	
Cenário Alternativo - Regra de dedução perigosa.	
Ações do AgentRoleActor	

1. Identificar que regra de dedução utilizada pelo aluno está correta, porém para esta questão pode resultar em um erro futuro.
2. Descrever aplicação de regra de dedução perigosa na questão do aluno.
3. Informar Mediador da anomalia encontrada na questão do aluno.

Nome do caso de uso interno	Selecionar questão na base de conhecimento
Stereotype	Plan
AgentRoleActor	Especialista Técnico
Resumo	Este caso de uso interno descreve as etapas seguidas pelo agente que assume a função de Especialista Técnico para selecionar uma questão para um aluno.
Cenário Principal	
Ações do AgentRoleActor	
1. Solicitar perfil do aluno ao Mediador	
2. Solicitar rede bayesiana do aluno ao Analista Bayesiano	
3. Executar caso de uso interno “Perceber perfil do aluno”	
4. Executar caso de uso interno “Perceber envio de rede bayesiana do aluno pelo Analista Bayesiano”.	
Cenário Alternativo - Envio de rede Bayesiana e de perfil do aluno Percebidos	
Ações do AgentRoleActor	
1. Verificar base de conhecimento	
2. Verificar histórico do aluno presente na rede bayesiana	
3. Filtrar as questões que não foram aplicadas anteriormente ao aluno, com base no histórico do aluno.	
4. Determinar a questão com base no nível do aluno presente em seu perfil	
5. Atribuir a complexidade da questão com base no histórico do aluno e nas informações presentes na questão.	

Nome do caso de uso interno	Determinar conteúdo
Stereotype	Plan
AgentRoleActor	Especialista Técnico
Resumo	Este caso de uso interno descreve as etapas seguidas pelo agente que assume a função de Especialista Técnico para selecionar uma questão para determinar o conteúdo de estudo do aluno.
Cenário Principal	
Ações do AgentRoleActor	

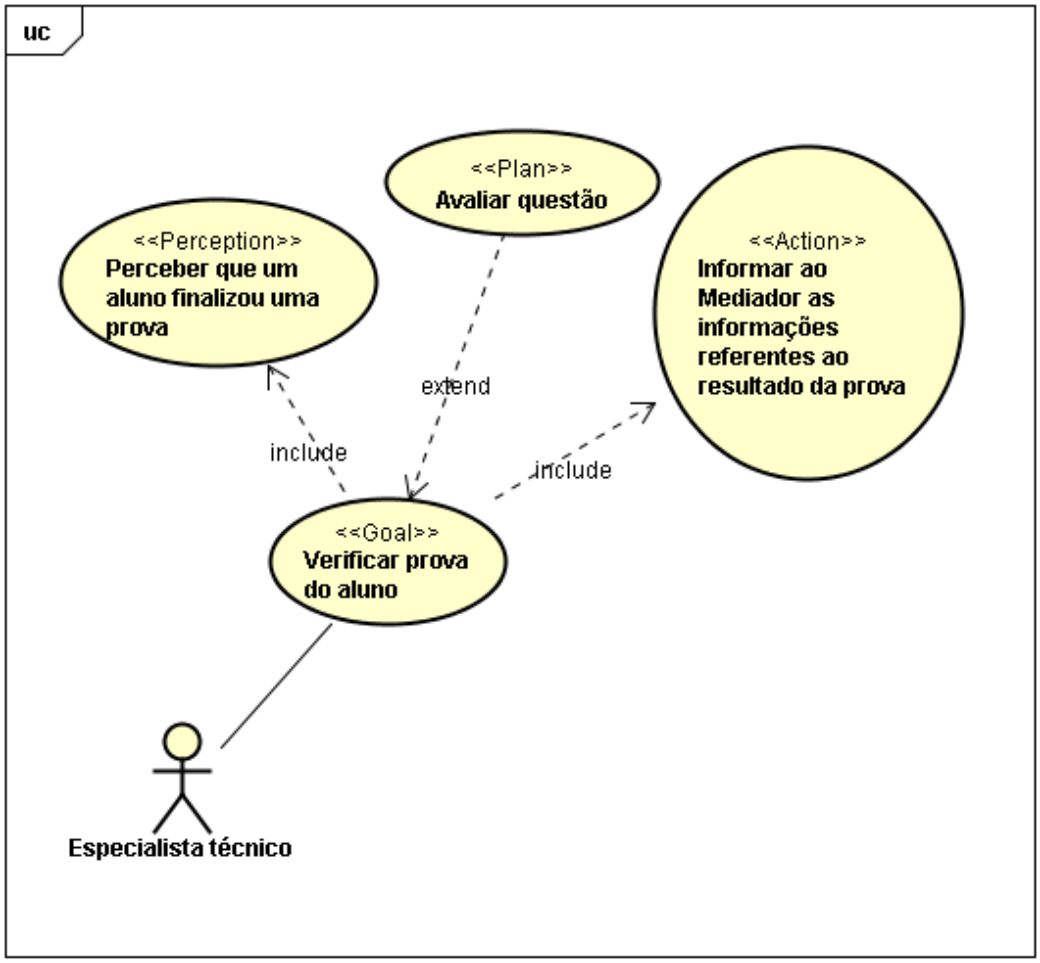
1. Solicitar o perfil do aluno ao Mediador
2. Solicitar rede bayesiana do aluno ao Analista Bayesiano
3. Executar caso de uso interno “Perceber perfil do aluno”
4. Executar caso de uso interno “Perceber envio de rede bayesiana do aluno pelo Analista Bayesiano”
Cenário Alternativo - Envio de rede Bayesiana e de perfil do aluno Percebidos
Ações do AgentRoleActor
1. Verificar base de conhecimento
2. Verificar histórico do aluno presente na rede bayesiana
3. Separar conteúdos que melhor se adequem ao perfil e ao histórico do aluno
4. Determinar sequência que beneficie melhor o aprendizado desses conteúdos
5. Enviar sequência de conteúdo ao Mediador

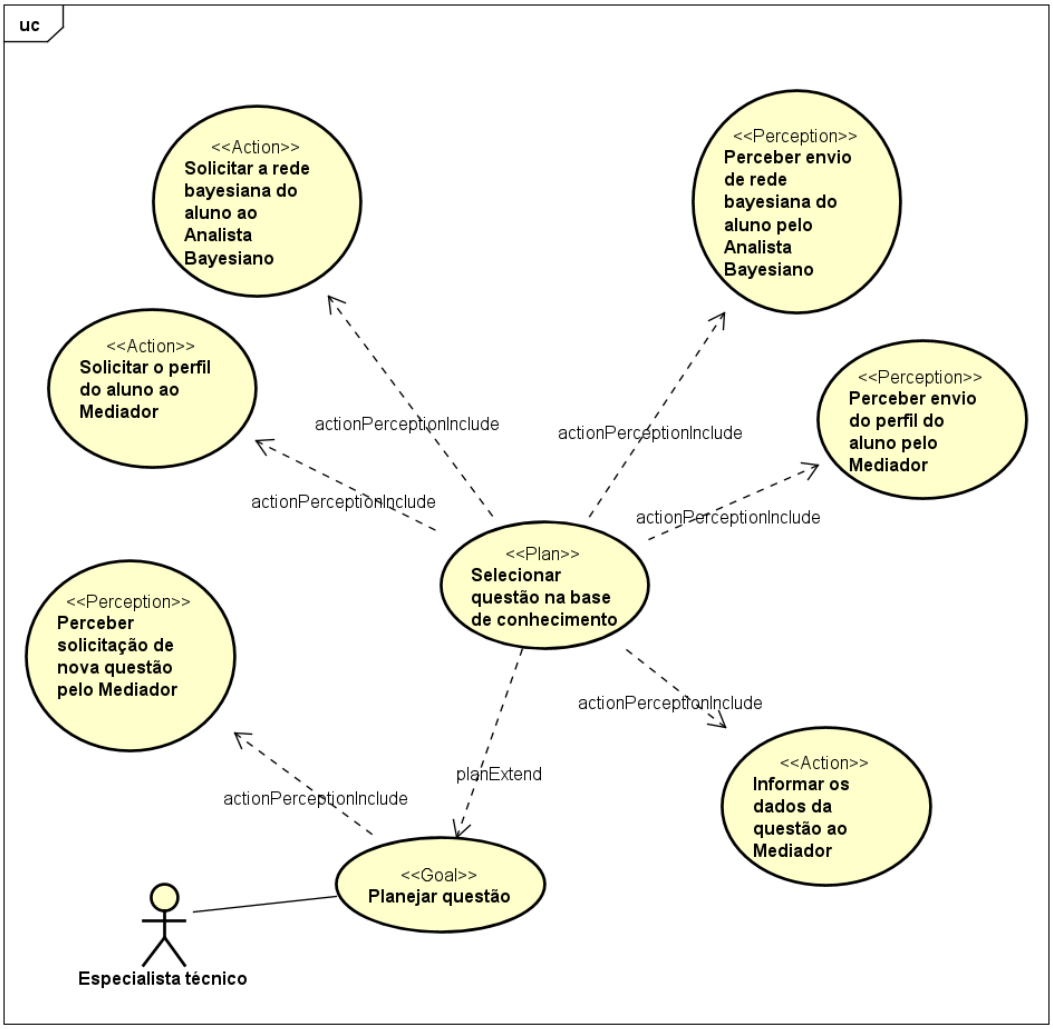
Nome do caso de uso interno	Preparar aplicação da questão
Stereotype	Plan
AgentRoleActor	Especialista Técnico
Resumo	Este caso de uso interno descreve as etapas seguidas pelo agente que assume a função de Mediador para solicitar a questão ao Especialista Técnico para possibilitar a aplicação da mesma.
Cenário Alternativo - Envio de nova questão do aluno percebida	
Ações do AgentRoleActor	
1. Aplicar questão ao aluno.	

3.12. Perspectivas de requisitos funcionais

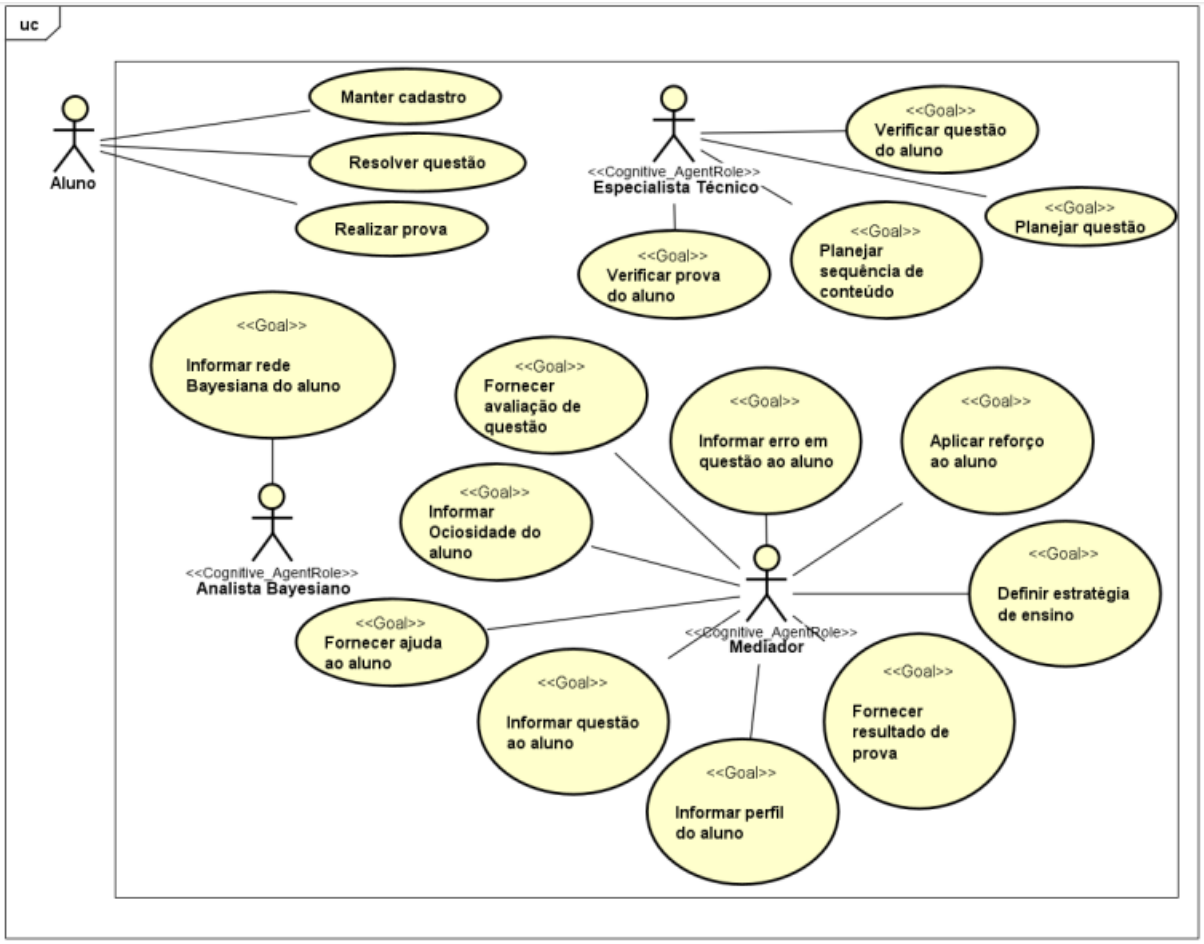
3.12.1. Diagrama de casos de uso internos de papel de agente

3.12.1.1. Papel de Agente Especialista Técnico


uc
<<Plan>>
Avaliar questão
<<Perception>>
Perceber que um aluno finalizou uma prova
<<Action>>
Informar ao Mediador as informações referentes ao resultado da prova
extend
include
include
<<Goal>>
Verificar prova do aluno
Especialista técnico


uc
<<Action>>
Solicitar a rede bayesiana do aluno ao Analista Bayesiano
<<Perception>>
Perceber envio de rede bayesiana do aluno pelo Analista Bayesiano
<<Action>>
Solicitar o perfil do aluno ao Mediador
<<Perception>>
Perceber envio do perfil do aluno pelo Mediador
actionPerceptionInclude
actionPerceptionInclude
actionPerceptionInclude
actionPerceptionInclude
<<Plan>>
Selecionar questão na base de conhecimento
<<Perception>>
Perceber solicitação de nova questão pelo Mediador
actionPerceptionInclude
<<Action>>
Informar os dados da questão ao Mediador
planExtend
actionPerceptionInclude
<<Goal>>
Planejar questão
Especialista técnico

3.12.2. Diagrama de casos de uso Geral



4. Verificação

4.1. Inspeção geral do documento de especificação de requisitos

4.2. Inspeção geral de cada casos de uso interno

4.3. Inspeção de cada caso de uso interno do tipo Goal

4.4. Inspeção de cada caso de uso interno do tipo Perception

4.5. Inspeção de cada diagrama de caso de uso parcial para cada AgentRoleActor

4.6. Inspeção do diagrama de casos de uso geral

5. Apêndices

5.1. Suposições e dependências

Não se aplica.

5.2. Acrônimos e abreviações

Descrever os acrônimos e abreviações utilizados no documento de Especificação de Requisitos de Software.

Acrônimo ou Abreviação	Descrição
uc	Caso de Uso ou Use Case

ANEXO B – TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO

Termo de Consentimento Livre e Esclarecido

Você está sendo convidado(a) para participar, como voluntário(a), de um experimento de inspeção de requisitos para sistemas multiagentes. Este estudo é parte integrante da pesquisa de mestrado realizada pelo aluno Giovane D'Avila Mendonça e coordenada pelo Prof. Dr. Gilleanes Thorwald Araujo Guedes.

Por meio deste termo, a qualquer momento, você poderá solicitar esclarecimentos adicionais sobre o estudo ou sobre a pesquisa. Também poderá retirar seu consentimento ou interromper a participação a qualquer momento, sem sofrer qualquer tipo de penalidade ou prejuízo. Ao participar deste estudo você não terá nenhum custo e nem receberá qualquer vantagem financeira. Seus dados pessoais serão mantidos em sigilo. Os resultados deste estudo serão armazenados pelo pesquisador responsável e poderão ser divulgados em publicações científicas, mantendo-se o anonimato dos dados pessoais.

Para participar deste estudo, você precisará: ouvir as explicações, ler o material fornecido, executar as atividades solicitadas e reportar sua avaliação. Ao participar deste estudo, você corre o risco de frustrar-se por não conseguir realizar as atividades solicitadas. Esperamos que os resultados deste estudo contribuam para garantir a qualidade da especificação de requisitos para sistemas multiagentes por meio da técnica de inspeção.

*Obrigatório

1. Endereço de e-mail *

meuemail@unipampa.edu.br

2. *

Marcar apenas uma oval.

- ◯ Tendo em vista as informações acima apresentadas, de forma livre e esclarecida, aceito participar deste estudo. *Pular para a pergunta 3*
- ◯ Não aceito.

Pular para a pergunta 3

Dados de identificação

Solicitamos que você insira alguns dados de identificação.

3. Matrícula *

4. Curso

Identificação de perfil do(a) voluntário(a)

Solicitamos que responda estas perguntas apenas para identificarmos o perfil dos voluntários(as).

5. Qual o seu nível de conhecimento em sistemas multiagentes? (considerando que, ao selecionar 0 você julga não ter nenhum conhecimento e 10 você tem total domínio do assunto) *

Marcar apenas uma oval.

0	1	2	3	4	5	6	7	8	9	10
◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯

6. Qual o seu nível de conhecimento em especificação de requisitos? (considerando que, ao selecionar 0 você julga não ter nenhum conhecimento e 10 você tem total domínio do assunto) *

Marcar apenas uma oval.

1	2	3	4	5	6	7	8	9	10
◯	◯	◯	◯	◯	◯	◯	◯	◯	◯

7. Qual o seu nível de conhecimento em Perspective-Based Reading ou Leitura Baseada em Perspectiva (PBR)? (considerando que, ao selecionar 0 você julga não ter nenhum conhecimento e 10 você tem total domínio do assunto) *

Marcar apenas uma oval.

1	2	3	4	5	6	7	8	9	10
◯	◯	◯	◯	◯	◯	◯	◯	◯	◯

8. Você já inspecionou requisitos na academia ou indústria? *

Marcar apenas uma oval.

- ◯ Sim
- ◯ Não

9. Neste momento, você está participando de outro experimento de inspeção de especificação de requisitos para sistemas multiagentes? *

Marcar apenas uma oval.

- ◯ Sim
- ◯ Não

ANEXO C – GIRSMA: GUIA DE INSPEÇÃO DE REQUISITOS PARA SISTEMAS MULTIAGENTES

GIRSMA - Guia de Inspeção de Requisitos para Sistemas Multiagentes

Autores
Giovane D'Avila Mendonça
Gilleanes Thorwald Araujo Guedes

1. Introdução

A validação de requisitos tem o objetivo de mostrar que os requisitos realmente definem o sistema desejado pelo usuário (SOMMERVILLE, 2007). A validação de requisitos é uma fase necessária da engenharia de requisitos e desempenha um papel crucial para que qualquer projeto seja bem sucedido. Ela visa garantir que as necessidades dos clientes sejam completas e bem escritas (GUPTA et al., 2019).

Na etapa de validação dos requisitos, qualquer artefato produzido pela engenharia de requisitos de determinado sistema deve ser avaliado quanto à qualidade (PRESSMAN, 2011).

A validação de requisitos garante que os requisitos escritos na especificação de requisitos de software sejam completos, consistentes e estão de acordo com o que o cliente precisa. Ela garante a validade dos requisitos do usuário eliminando ambiguidades e inconsistências da especificação de requisitos de software (BILAL et al., 2016).

Várias técnicas de validação de requisitos foram aplicadas para desenvolver software de qualidade, como revisão de requisitos, inspeções, prototipagem, baseado em modelo, teste de requisitos e validação de requisitos orientada por ponto de vista (SAQI; AHMED, 2008; RAJA, 2009; YOUSUF et al., 2008).

Entre estas técnicas, as inspeções de software são um meio eficiente de melhorar a qualidade (CARNEIRO et al., 2017). E a técnica de inspeção de Leitura Baseada em Perspectiva ou *Perspective-Based Reading* (PBR) é uma das técnicas mais eficazes para inspeção de requisitos (EBAD, 2017).

Desta forma, este guia compreende uma adaptação da técnica PBR (BASILI et al., 1996; SHULL et al., 2000) para o contexto de sistemas multiagentes. Para isso, é proposta uma nova perspectiva denominada de PBR-Simulador de Agente. Esta perspectiva procura determinar se os requisitos descrevem adequadamente as funcionalidades executadas por agentes. Esta perspectiva inspeciona os Casos de Uso Internos (IUC), observando se os cenários principais e cenários alternativos estão descritos, quando houver se as pré-condições estão definidas, entre outras informações.

2. Definições e Termos utilizados

- Papéis de agentes: representam as funções que os agentes podem desempenhar dentro de um sistema. Os agentes podem assumir mais de um papel, mas geralmente não ao mesmo tempo.
- Verbo de ação: indica uma ação, acontecimento, desejo ou atividade mental. O próprio verbo exprime algo que será feito pelo sujeito. Por exemplo: Adaptar, Verificar, Informar.
- Estereótipo: os estereótipos (*Stereotype*) são um mecanismo de extensibilidade da UML (UML, 2021). Eles dão mais poder à UML, permitindo classificar elementos com algo em comum.
- Caso de Uso Interno: os Casos de Uso Internos ou *Internal Use Case* (IUC) são casos de uso que não são acessados por atores externos, eles se diferenciam dos casos de uso normais por representarem funcionalidades internas que não podem ser acessadas pelos usuários (eles sequer têm consciência de sua existência). IUCs são acessados por Atores de Papéis de Agente (*AgentRoleActors*). Já os casos de uso normais são acessados por atores externos (fora da fronteira do sistema representados por atores normais).
- *AgentRoleActors:* representam papéis assumidos por agentes que podem ter estereótipos do tipo ReactiveAgentRoles ou CognitiveAgentRoles, esses papéis de agentes são modelados dentro da fronteira do sistema.

3. Escopo

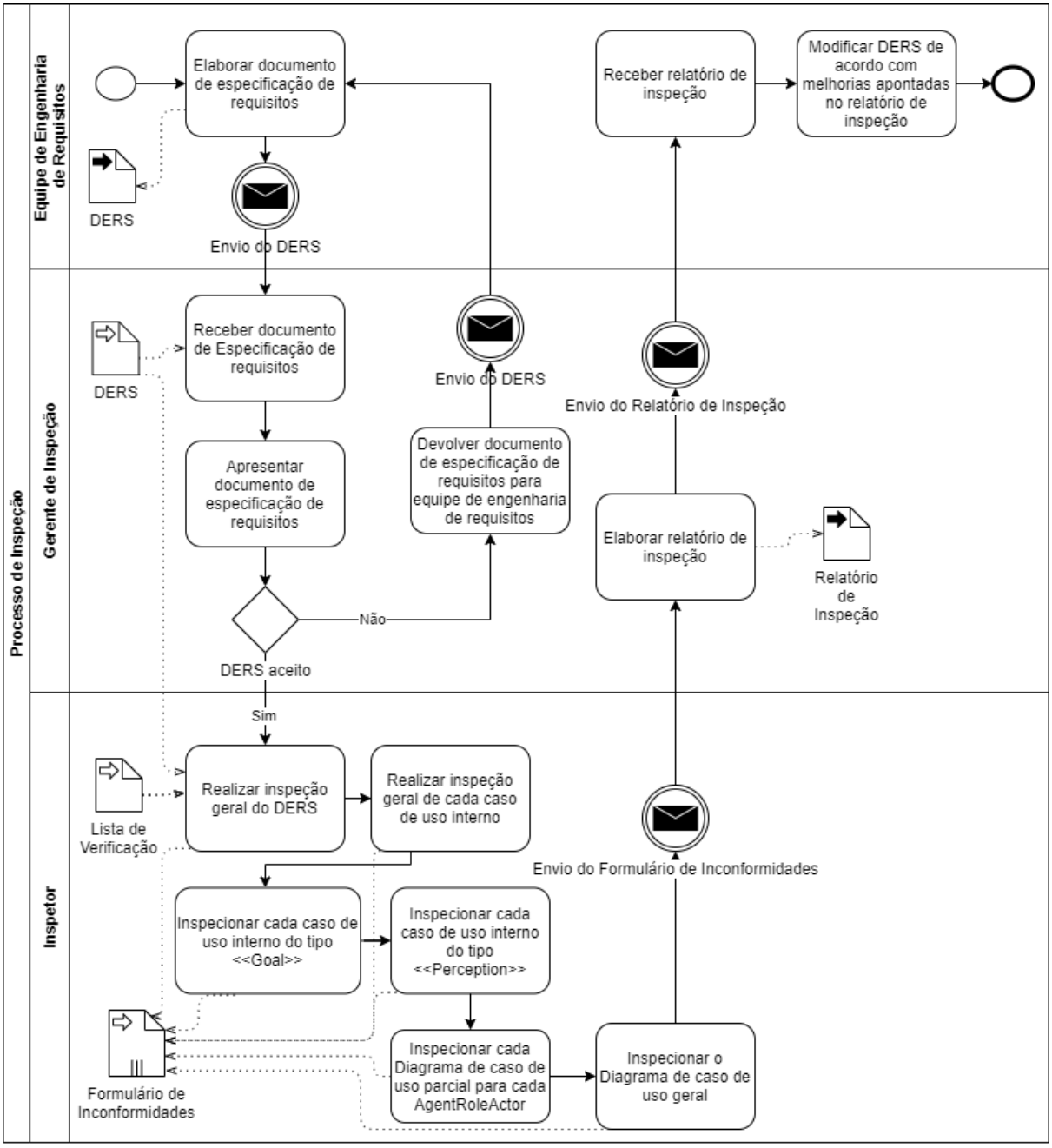
O método de desenvolvido para validação do documento de especificação de requisitos de software, em constante processo evolutivo, é denominado de GIRSMA - Guia de Inspeção de Requisitos para Sistemas Multiagentes. Este método foi desenvolvido com base na técnica de Leitura Baseada em Perspectiva ou *Perspective-Based Reading* (PBR) (BASILI et al., 1996; SHULL et al., 2000).

Este método visa garantir um documento de especificação de requisitos de software (DERS) completo, correto, consistente e de fácil entendimento. Esta versão do GIRSMA visa inspecionar um DERS produzido com base em uma extensão do template do padrão ISO/IEC/IEEE 29148:2018 (ISO/IEC/IEEE 29148:2018, 2018) e na notação da MASRML - *Multiagent Systems Requirements Modelling Language* (Guedes et al., 2020). Esta notação utiliza conceitos como por exemplo Casos de Uso Internos ou *Use Case Internal* (IUC) e Atores de Papéis de Agente ou *AgentRoleActor*.

4. Estrutura do GIRSMA

O guia de inspeção de requisitos para sistemas multiagentes é composto por um processo de inspeção. Este processo compreende um conjunto de atividades que devem ser realizadas para garantir a qualidade do documento de especificação de requisitos de software. O processo pode ser observado na Figura 1.

Figura 1 - Processo de inspeção para sistemas multiagentes.



Conforme pode ser observado na Figura 1, o inspetor deve realizar uma sequência de seis passos. Para realizar estes passos serão utilizadas Listas de Verificações como guias para inspecionar partes do DERS. Os seis passos compreendem à:

- Primeiro passo: Inspeção geral do documento de especificação de requisitos

Nesta etapa, o documento de especificação de requisitos é inspecionado de forma geral. O inspetor buscará falhas do tipo completude, corretude e consistência. Algumas delas estão relacionadas à definição de papéis de agentes, as funcionalidades associadas aos agentes, padrão de comunicação utilizado e sobre as crenças identificadas. Apresentamos na Figura 2, a Lista de Verificação para a inspeção geral do documento de especificação de requisitos.

Figura 2 - Lista de verificação do primeiro passo.

Nº	Questão	Regra	Opção	
1.1	Os papéis de agente para o sistema foram definidos?	Completude	Sim ( )	Não ( )*
1.2	As descrições dos papéis de agente estão claras?	Corretude	Sim ( )	Não ( )*
1.3	Foram atribuídas funcionalidades aos papéis de agente?	Consistência	Sim ( )	Não ( )*
1.4	As funcionalidades atribuídas ao papel de agente estão claras?	Corretude	Sim ( )	Não ( )*
1.5	Existe alguma forma de comunicação entre os agentes e usuários externos?	Completude	Sim ( )	Não ( )*
1.6	Foi estabelecido qual o tipo de informação a ser transmitida entre os agentes e usuários externos?	Consistência	Sim ( )	Não ( )*
1.7	Existe alguma forma de comunicação entre os agentes?	Completude	Sim ( )	Não ( )*
1.8	Foi definido o padrão de comunicação dos agentes?	Completude	Sim ( )	Não ( )*
1.9	Foi estabelecido qual o tipo de informação a ser transmitida entre os agentes?	Consistência	Sim ( )	Não ( )*
1.10	A crença identificada está atribuída a um objetivo?	Completude	Sim ( )	Não ( )*
1.11	A descrição da crença está clara?	Consistência	Sim ( )	Não ( )*
1.12	A descrição da crença está duplicada?	Consistência	Sim ( )	Não ( )*
1.13	A descrição da crença está conflitante com outra crença?	Consistência	Sim ( )	Não ( )*

* Informe no Formulário de Inconformidades.

- Segundo passo: Inspeção geral de cada caso de uso interno

Nesta etapa, o inspetor deve analisar cada IUC buscando falhas do tipo corretude, não ambiguidade, rastreabilidade, completude e consistência. Algumas destas falhas estão relacionadas à estrutura geral de cada IUC, falta de cenário principal, ambiguidades ou corretude dos textos dos casos de uso.

Apresentamos na Figura 3, a Lista de Verificação para a inspeção de cada IUC descrito no documento de especificação de requisitos.

Figura 3 - Lista de verificação do segundo passo.

Nº	Questão	Regra	Opção	
2.1	O texto do caso de uso interno apresenta erros de português?	Corretude	Sim ( )*	Não ( )
2.2	O texto do caso de uso interno está escrito de forma ambígua?	Ambiguidade	Sim ( )*	Não ( )
2.3	O caso de uso interno possui um código para identificação?	Rastreabilidade	Sim ( )	Não ( )*
2.4	O nome do caso de uso interno está escrito com verbo de ação?	Corretude	Sim ( )	Não ( )*
2.5	Os papéis de agente (*AgentRoleActor*) associados ao caso de uso interno estão identificados?	Completude	Sim ( )	Não ( )*
2.6	O papel do agente que interage com o caso de uso interno está descrito na seção adequada?	Completude	Sim ( )	Não ( )*
2.7	O caso de uso interno apresenta resumo que descreve de forma clara o propósito do caso de uso interno?	Completude	Sim ( )	Não ( )*
2.8	Se o caso de uso interno possuir crenças iniciais, estas crenças entram em conflito com outras crenças?	Consistência	Sim ( )*	Não ( )
2.9	Se a Pré-condição exige a execução de outro caso de uso interno, este está descrito separadamente?	Consistência	Sim ( )	Não ( )*
2.10	O caso de uso interno tem um cenário principal?	Corretude	Sim ( )	Não ( )*
2.11	Se um cenário principal exige a execução de outro caso de uso interno, este está descrito separadamente?	Consistência	Sim ( )	Não ( )*
2.12	Se um cenário alternativo exige a execução de outro caso de uso interno, este está descrito separadamente?	Consistência	Sim ( )	Não ( )*

* Informe no Formulário de Inconformidades.

- Terceiro passo: Inspeção de cada caso de uso interno do tipo Goal

Nesta etapa, o inspetor deve analisar cada um dos casos de uso internos do tipo Goal, buscando falhas do tipo corretude, consistência e completude. Algumas delas estão relacionadas à corretude do estereótipo, clareza na descrição das percepções que tornam o objetivo uma intenção ou existência de percepções associadas ao objetivo. Apresentamos na Figura 4, a Lista de Verificação para a inspeção de cada IUC do tipo Goal.

Figura 4 - Lista de verificação do terceiro passo.

Nº	**Questão**	**Regra**	**Opção**	
3.1	Casos de uso internos que representam desejo (*Goal*-Objetivo) dos papéis de agentes estão identificados com esterótipo do tipo *Goal*?	Corretude	Sim ( )	Não ( )*
3.2	A(s) percepção(ões) que torna(m) o objetivo (*Goal*) uma intenção está(ão) descrita(s) de forma clara?	Corretude	Sim ( )	Não ( )*
3.3	É possível estabelecer critérios de prioridades de desejos (*goals*/objetivos) a serem atingidos/satisfeitos?	Consistência	Sim ( )	Não ( )*
3.4	É necessário um cenário alternativo para o objetivo (*Goal*) se tornar uma intenção?	Completude	Sim ( )	Não ( )*
3.5	A(s) crença(s) inicial(ais) esta(ão) descrita(s)?	Completude	Sim ( )	Não ( )*
3.6	Existem percepções associadas ao objetivo?	Completude	Sim ( )	Não ( )*
3.7	Está claro como o objetivo se torna uma intenção?	Consistência	Sim ( )	Não ( )*
3.8	Existe um cenário em caso de falha na execução de uma intenção?	Consistência	Sim ( )	Não ( )*

* Informe no Formulário de Inconformidades.

- Quarto passo: Inspeção de cada caso de uso interno do tipo Perception

Nesta etapa, o inspetor analisará cada um dos casos de uso internos do tipo Perception, buscando falhas do tipo corretude e completude. Algumas delas estão relacionadas à corretude do estereótipo, identificação e clareza das crenças iniciais e definição de pré-condições. Apresentamos na Figura 5, a Lista de Verificação para a inspeção de cada IUC do tipo Perception.

Figura 5 - Lista de verificação do quarto passo.

Nº	Questão	Regra	Opção	
4.1	O estereótipo do caso de uso interno foi definido como Perception?	Corretude	Sim ( )	Não ( )*
4.2	As Pré-condições estão definidas?	Completude	Sim ( )	Não ( )*
4.3	As crenças iniciais estão descritas?	Completude	Sim ( )	Não ( )*
4.4	As crenças iniciais estão descritas de forma clara?	Corretude	Sim ( )	Não ( )*

* Informe no Formulário de Inconformidades.

- Quinto passo: Inspeção de cada diagrama de caso de uso parcial para cada *AgenRoleActor*

Nesta etapa o inspetor deve analisar cada diagrama de caso de uso parcial elaborado para cada *AgentRoleActor*. Ele buscará falhas do tipo corretude e consistência. Algumas delas estão relacionadas à corretude dos estereótipos, corretude da associação entre componentes do diagrama e a corretude da direção das associações entre componentes do diagrama. Apresentamos na Figura 6, a Lista de Verificação para inspeção de cada Diagrama de caso de uso parcial para cada *AgentRoleActor.*

Figura 6 - Lista de verificação do quinto passo.

Nº	Questão	Regra	Opção	
5.1	Os objetivos (*Goals*) associados ao papel de agente estão identificados com o estereótipo do tipo *Goal*?	Corretude	Sim ( )	Não ( )*
5.2	Se houver associação entre um caso de uso interno Objetivo (*Goal*) e um caso de uso interno de ação (*Action*), esta associação é do tipo *actionPerceptionInclude*?	Corretude	Sim ( )	Não ( )*
5.3	A direção da associação é do objetivo (*Goal*) para a ação (*Action*)?	Corretude	Sim ( )	Não ( )*
5.4	Se houver associação entre um caso de uso interno Objetivo (*Goal*) e um caso de uso interno de percepção (*Perception*) esta associação é do tipo *actionPerceptionInclude*?	Completude	Sim ( )	Não ( )*
5.5	Os casos de uso internos do tipo percepção (*Perception*) estão identificados com o estereótipo *Perception*?	Corretude	Sim ( )	Não ( )*
5.6	A direção da associação é do objetivo (*Goal*) para o caso de uso interno do tipo *Perception*?	Corretude	Sim ( )	Não ( )*
5.7	O plano (*Plan*) está identificado com o estereótipo do tipo *Plan*?	Corretude	Sim ( )	Não ( )*
5.8	O plano (*Plan*) está associado a pelo menos um caso de uso interno do tipo *Goal*?	Consistência	Sim ( )	Não ( )*
5.9	A associação do Plano com o caso de uso interno Objetivo (*Goal*) é do tipo *planExtend*?	Corretude	Sim ( )	Não ( )*
5.10	A direção da associação é do plano (*Plan*) para o caso de uso interno do tipo *Goal*?	Corretude	Sim ( )	Não ( )*
5.11	A associação do plano (*Plan*) com o caso de uso interno do tipo *Action* é do tipo *actionPerceptionInclude*?	Corretude	Sim ( )	Não ( )*
5.12	A associação do plano com a percepção é do tipo *actionPerceptionInclude*?	Corretude	Sim ( )	Não ( )*
5.13	A direção da associação é do plano (*Plan*) para a ação (*Action*)?	Corretude	Sim ( )	Não ( )*
5.14	A ação (*Action*) está identificada com estereótipo do tipo *Action*?	Corretude	Sim ( )	Não ( )*
5.15	A ação (*Action*) está associada a pelo menos um caso de uso interno do tipo *Plan* ou *Goal*?	Consistência	Sim ( )	Não ( )*

* Informe no Formulário de Inconformidades.

- Sexto passo: Inspeção do diagrama de caso de uso geral

Nesta etapa o inspetor deve analisar o diagrama de caso de uso geral, buscando localizar falhas do tipo corretude e completude. Algumas delas estão relacionadas à corretude do estereótipo dos papéis de agentes, papéis de agentes e usuários externos identificados nos locais corretos, e atribuição de pelo menos um objetivo a cada papel de agente. A Lista de Verificação para inspeção do Diagrama de Caso de Uso geral pode ser verificada na Figura 7.

Figura 7 - Lista de verificação do sexto passo.

Nº	Questão	Regra	Opção	
6.1	Os atores que representam os usuários externos estão identificados fora da fronteira do sistema (representada por um retângulo)?	Corretude	Sim ( )	Não ( )*
6.2	Os papéis de agentes estão identificados dentro da fronteira do sistema (retângulo)?	Corretude	Sim ( )	Não ( )*
6.3	Os papéis de agentes estão identificados com estereótipos do tipo ReactiveAgentRole, CognitiveAgentRole, AgentRolePS, AgentRoleUAM ou AgentRoleSMI?	Corretude	Sim ( )	Não ( )*
6.4	Existe pelo menos um objetivo associado a cada papel de agente?	Completude	Sim ( )	Não ( )*

* Informe no Formulário de Inconformidades.

- Formulário de inconformidades

Se o inspetor acredita ter encontrado uma falha ou problema que exija atenção da equipe de requisitos, deve anotar esta falha ou problema no Formulário de Inconformidades. Este formulário descreve, entre outras informações, a localização e o tipo da falha. A Tabela 1 apresenta a estrutura do formulário.

Tabela 1 - Formulário de Inconformidades.

FORMULÁRIO DE INCONFORMIDADES	
Doc. de Especificação de Requisitos (código, nome, versão):	
Nome Arquivo:	
Elaborador (papel, nome, e-mail):	
Data de Revisão (DD/MM/AAAA): _____/_____/_____	Tempo de Revisão (HH:MM): _____:_____

Nº Questão	Página	Linha	*Tipo	Descrição
<1.5>	<00>	<00>	<ICM>	
OBSERVAÇÕES GERAIS DOS REVISORES				
1. 2. 3.				
*Sigla da Taxonomia: ICM- Incompleto, ICR-Incorreto, ICS-Inconsistente, NV-Não verificável, ONL-Outro não listado.				

5. Referências

BASILI, V. R.; GREEN, S.; LAITENBERGER, O.; LANUBILE, F.; SHULL, F.; SORUMGARD, S. and ZELKWITZ, M. V., “The empirical investigation of perspective-based reading,” Empirical Software Engineering, vol. 1, pp.133–164, 1996.

BILAL, H. A. et al. Requirements validation techniques: An empirical study.International Journal of Computer Applications, Foundation of ComputerScience (FCS), NY, USA, New York, USA, v. 148, n. 14, p. 5–10, Aug 2016. ISSN 0975-8887.

CARNEIRO, G. et al. Investigating the influence of inspector learning styles on design inspections: Findings of a quasi-experiment.XX Ibero-American Conference on Software Engineering, 2017.

EBAD, S. A. Inspection reading techniques applied to software artifacts - a systematic review. Comput. Syst. Sci. Eng., v. 32, 2017.

GUEDES, G. SOUZA FILHO, I. GAEDICKE, L. MENDONÇA, G. VICARI, R. and BRUSIUS, C. “Masrml - a domain-specific modeling language for multi-agent systems requirements,” International Journal of Software Engineering Applications (IJSEA), vol. 11, no. 5, 2020.

GUPTA, A.; DERAMAN, A.; SIDDIQUI, S. Bdi logics for bdi architectures: old problems, new perspectives. ARPN Journal of Engineering and Applied Sciences, Asian Research Publishing Network, v. 14, n. 17, p. 3046–3061, 2019.

ISO/IEC/IEEE 29148. “Iso/iec/ieee international standard - systems and software engineering – life cycle processes – requirements engineering,” ISO/IEC/IEEE 29148:2018(E), pp. 1–104, 2018.

PRESSMAN, R.Engenharia de Software: Uma Abordagem Profissional. [S.l.]:Bookman, 2011.

RAJA, U. A. Empirical studies of requirements validation techniques. 2nd International Conference on Computer Blekinge Institute of Technology, 2009.

SAQI, S. B.; AHMED, S. Requirements validation techniques practiced in industry: Studies of six companies.Blekinge Institute of Technology, 2008

SHULL, F.; RUS, I.; BASILI, V. How perspective-based reading can improve requirements inspections. Computer, v. 33, n. 7, p. 73–79, 2000.

SOMMERVILLE, I.Engenharia de Software. [S.l.]: Addison Wesley, 2007.

UML 2.4.1 Superstructure Specification. 2021. Disponível em: <https://www.omg.org/spec/UML/2.5/About-UML/>.

YOUSUF, F.; ZAMAN, Z.; IKRAM, N. Requirements validation techniques in gsd: Asurvey.IEEE International Multitopic Conference, 2009.

ANEXO D – FORMULÁRIO DE INCONFORMIDADES

Você deve informar neste formulário todas as inconformidades encontradas com base nas listas de verificações.

FORMULÁRIO DE INCONFORMIDADES	
Doc. de Especificação de Requisitos (código, nome, versão):	
Nome Arquivo:	
~~Elaborador (papel, nome, e-mail):~~	
Data de Revisão (DD/MM/AAAA): _____/_____/_____	Tempo de Revisão (HH:MM): _____:_____

INCONFORMIDADES ENCONTRADAS				
Nº Questão	Página	Linha	*Tipo	Descrição
01	00	00	ICM	

OBSERVAÇÕES GERAIS DOS REVISORES
1. 2. 3.
*Sigla da Taxonomia: ICM- Incompleto, ICR-Incorreto, ICS-Inconsistente, NV-Não verificável, ONL-Outro não listado.

ÍNDICE