UNIVERSIDADE FEDERAL DO PAMPA

Iderli Pereira de Souza Filho

Uma Proposta de Incorporação do Diagrama de Classes da UML à Linguagem MASRML Adaptado ao Contexto de Sistema Multiagentes

Alegrete
2023

Iderli Pereira de Souza Filho

Uma Proposta de Incorporação do Diagrama de Classes da UML à Linguagem MASRML Adaptado ao Contexto de Sistema Multiagentes

Dissertação 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
2023

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) .

F478678p Filho, Iderli Pereira de Souza
Uma Proposta de Incorporação do Diagrama de Classes da UML à Linguagem MASRML Adaptado ao Contexto de Sistema Multiagentes / Iderli Pereira de Souza Filho.
115 p.

Dissertação(Mestrado)-- Universidade Federal do Pampa, MESTRADO EM ENGENHARIA DE SOFTWARE, 2023.
"Orientação: Gilleanes Thorwald Araujo Guedes".

1. UML. 2. Diagrama de Classes. 3. Sistema Multiagente. 4. Linguagens de Modelagem. I. Título.

Iderli Pereira de Souza Filho

Uma Proposta de Incorporação do Diagrama de Classes da UML à Linguagem MASRML Adaptado ao Contexto de Sistema Multiagentes

Dissertação 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.

Dissertação defendida e aprovado em 19/06/2023

Banca examinadora:

Prof. Dr. Gilleanes Thorwald Araujo Guedes
Orientador
(Unipampa)

Profª. Dra. Rosa Maria Vicari
(UFRGS)

Prof. Dr. Maicon Bernardino da Silveira
(Unipampa)

Assinado eletronicamente por **GILLEANES THORWALD ARAUJO GUEDES**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 19/06/2023, às 15:24, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **ROSA MARIA VICARI**, **Usuário Externo**, em 19/06/2023, às 15:25, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **MAICON BERNARDINO DA SILVEIRA**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 19/06/2023, às 15:25, 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 **1155876** e o código CRC **37CD8005**.

Universidade Federal do Pampa, Campus Alegrete
Av. Tiarajú, 810 – Bairro: Ibirapuitã – Alegrete – RS CEP: 97.546-550
Telefone: (55) 3422-8400

Este trabalho é dedicado à todos que me apoiaram durante seu desenvolvimento.

AGRADECIMENTOS

Agradeço primeiramente à minha família, em especial aos meus irmãos, que estão sempre ao meu lado me apoiando, e aos meus pais, que são uma presença constante em todos os momentos da minha vida. Sei que posso sempre contar com vocês e tenho um imenso orgulho disso.

Gostaria de expressar minha profunda gratidão ao meu orientador, o Prof. Dr. Gilleanes Thorwald Araujo Guedes, por ter aceitado me orientar mesmo à distância. Sua disponibilidade em tirar minhas dúvidas e as correções minuciosas que realizou no texto foram essenciais para a conclusão deste trabalho. Sem a sua orientação, seria impossível alcançar esse resultado.

Também desejo agradecer aos meus queridos colegas e amigos, que generosamente dedicaram seu tempo e esforço para participar do experimento de avaliação da notação. A contribuição valiosa de cada um de vocês foi fundamental para que eu pudesse atingir os resultados desejados.

Por fim, não posso deixar de mencionar minha gratidão à Universidade Federal do Pampa (UNIPAMPA) por proporcionar um ensino de qualidade de forma gratuita. Sou imensamente grato pela oportunidade de estudar em uma instituição que valoriza a educação e contribui para o meu crescimento acadêmico e pessoal.

Obrigado a todos!

Há alguns dias atrás vendi minha alma a um velho apache
Não é que eu ache que o mundo tenha salvação
Mas como diria o intrépido cowboy, fitando o bandido indócil
A alma é o segredo, a alma é o segredo
A alma é o segredo do negócio
(Baladas do Asfalto, Zeca Baleiro)

RESUMO

Na Engenharia de Software, a utilização de linguagens de modelagem, técnicas, métodos e processos de desenvolvimento auxiliam o projetista a desenvolver o software com maior qualidade. Dentre as linguagens de modelagem utilizadas no desenvolvimento de software, uma das mais conhecidas é a UML - Unified Modeling Language - e seu diagrama de classes é um dos mais utilizados, pois é capaz de demonstrar a estrutura lógica de softwares orientados a objetos por meio da representação de suas classes, atributos, operações e relações. Já na área de Inteligência Artificial Distribuída, os agentes de software - programas de computador caracterizados por possuírem autonomia, habilidade social, reatividade e proatividade - possuem uma estrutura e requisitos únicos, os quais não podem ser representados por meio do diagramas de classe padrão da UML. Sendo assim, é necessário adaptar este diagrama para o contexto de Sistemas Multiagentes. Tendo em mente esses pontos, percebemos a necessidade de verificar a existência de extensões do diagrama de classes da UML para o contexto de Sistemas Multiagentes e avaliar a sua real utilidade e suporte ao desenvolvimento deste tipo de software. Dessa forma, realizamos uma revisão sistemática de literatura, visando identificar os pontos fortes e fracos dessas extensões. Deste modo, analisamos as extensões encontradas na revisão e, após aplicar os diagramas apresentados, percebemos a necessidade de propor uma nova extensão e adaptação do diagrama de classes da UML adicionando-a à linguagem MASRML - Multi-Agent Systems Requirements Modeling Language - uma linguagem derivada da UML exclusiva para a representação de requisitos particulares de sistemas multiagentes. A extensão foi realizada com o objetivo de permitir que a linguagem MASRML suporte as características estruturais do diagrama de classes, devidamente adaptado ao contexto de sistemas multiagentes, de tal maneira que ela possa representar conceitos que consideramos importantes com base na revisão desenvolvida. Após produzir uma versão inicial, avaliamos nossa proposta através de um experimento baseado em comparação de modelos.

Palavras-chave: UML. Diagrama de Classes. Sistema Multiagente. Linguagens de Modelagem.

ABSTRACT

In Software Engineering, the use of modeling languages, techniques, methods, and development processes aids the designer to develop software with higher quality. Among the modeling languages used in software development, one of the best known is the UML – Unified Modeling Language – and its class diagram is one of the most used, as it is capable of demonstrating the logical structure of object-oriented software by means of the representation of their classes, attributes, operations, and relations. Now in the Distributed Artificial Intelligence area, software agents – computer programs characterized by having autonomy, social ability, reactivity, and proactivity – possess a unique structure and requirements, which cannot be represented through the standard UML class diagram. Thus, it is necessary to adapt this diagram to the Multiagent Systems context. Keeping this in mind, we realized the need to verify the existence of UML class diagram extensions for the context of multi-agent systems and evaluate their real usefulness and support for the development of this type of software. Thus, we carried out a systematic literature review, aiming to identify the strengths and weaknesses of these extensions. This way, we analysed the extensions found in the review and, after applying the diagrams presented, we realized the need to propose a new extension and adaptation of the UML class diagram, adding it to the MASRML (Multi-Agent Systems Requirements Modeling Language) – a language derived from UML exclusive to the representation of particular requirements of multiagent systems. The extension was carried out with the objective of allowing the MASRML language to support the structural characteristics of the class diagram, duly adapted to the multiagent systems context, in such a way that this language can represent concepts that we consider important based on the review conducted. After producing an initial version, we evaluated our proposal through an experiment based on comparing models.

Key-words: UML. Class Diagram, Multiagent System. Modeling Languages.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

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

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

MASRML *Multi-Agent Systems Requirements Modeling Language* ou Linguagem de Modelagem de Requisitos de Sistemas Multiagentes

OA Objeto de Aprendizagem

REPMAS *Requirements Engineering Process for MultiAgent Systems* ou Processo de Engenharia de Requisitos para Sistemas Multiagente

RSL Revisão Sistemática de Literatura

SMA Sistema Multiagente

SWEBOK *Software Engineering Body of Knowledge* ou Guia para o Corpo de Conhecimento em Engenharia de Software

UML *Unified Modeling Language* ou Linguagem de Modelagem Unificada

SUMÁRIO

1 INTRODUÇÃO

A tecnologia de agentes é um paradigma de software que fornece abstrações de agentes para o desenvolvimento de sistemas abertos, distribuídos e heterogêneos (GAN et al., 2020). Os Agentes de Software se caracterizam por serem autônomos, possuírem habilidades sociais, reatividade e pró-atividade. A autonomia é a capacidade do agente de operar sem intervenção direta do homem. A habilidade social permite aos agentes interagir entre si e, possivelmente, com usuários que estão utilizando o sistema, por meio de algum tipo de linguagem de comunicação de agente. A reatividade refere-se à capacidade do agente de perceber o ambiente e responder em tempo hábil às mudanças que ocorrem nele. E, por último, a pró-atividade é a capacidade de exibir um comportamento direcionado a um objetivo, tomando a iniciativa, sem receber ordens para isso (HAJDUK; SUKOP; HAUN, 2019).

Um Sistema Multiagente (SMA) é um tipo específico de sistema composto de vários agentes que interagem entre si para atingir determinados objetivos. Esses sistemas podem ser usados para resolver problemas que são difíceis ou impossíveis para um sistema monolítico ou para um único agente resolver (VIRUEL, 2011).

Os SMAs estão cada vez mais próximos de áreas críticas, como medicina, veículos autônomos, justiça criminal e mercados financeiros (CALVARESI et al., 2019). Na mesma linha, com o atual avanço da tecnologia, os aplicativos baseados em agentes estão se tornando um padrão em uma grande variedade de domínios, como comércio eletrônico, logística, gerenciamento de cadeia de suprimentos, telecomunicações, saúde e manufatura. Este avanço se deve ao fato de que esses sistemas são vistos como uma tecnologia e uma ferramenta que auxilia na análise e desenvolvimento de novos modelos e teorias em sistemas distribuídos de grande escala ou em sistemas centrados no homem (BOTTI et al., 2019).

É importante destacar também que existem diversos modelos ou arquiteturas para desenvolver um SMA, entre esses modelos podemos citar o modelo *Belief-Desire-Intention* ou crença–desejo–intenção (BDI). Neste modelo, de acordo com Herzig et al. (2017), os conceitos de crença e objetivo (desejo) desempenham um papel central na concepção e implementação de agentes autônomos. Bordini, Hübner e Wooldridge (2007) afirmam que os agentes BDI são adequados a cenários imprevisíveis que requerem decisões dinâmicas devido à sua capacidade de escolher um plano para atingir um objetivo, dadas as suas crenças.

No entanto, o desenvolvimento de SMAs apresenta sua própria complexidade e seus próprios desafios, o que demonstra a necessidade de aplicar práticas de Engenharia de Software no desenvolvimento desses sistemas. Isso levou ao surgimento da *Agent-Oriented Software Engineering* ou Engenharia de Software Orientada a Agentes (AOSE), uma área que mistura características da Engenharia de Software e da Inteligência Artificial.

Segundo (GUEDES; VICARI, 2010), esta área tem como objetivo adaptar as práticas de Engenharia de Software especificamente para o desenvolvimento de sistemas baseados em agentes.

Dentro deste contexto, Bauer e Odell (2005) afirmam que a implantação de tecnologia de agentes com sucesso em aplicações industriais requer métodos de software de qualidade industrial e ferramentas de engenharia explícitas, como a *Unified Modeling Language* ou Linguagem de Modelagem Unificada (UML), a linguagem-padrão de modelagem adotada internacionalmente pela indústria de engenharia de software (GUEDES, 2009). Um dos diagramas mais conhecidos da UML é o diagrama de classes, que desempenha um papel importante no processo de desenvolvimento de software. O diagrama de Classes permite representar a estrutura lógica do sistema, baseando-se no paradigma da orientação a objetos. No entanto, é necessário um grande esforço para conduzir o processo de diagramação de Classes com precisão, para evitar falhas do sistema no futuro (SADOWSKA, 2020).

Considerando que a UML padrão não possui mecanismos para diagramar SMAs e não consegue modelar conceitos como percepções, objetivos, crenças e planos, entre outras características de agentes, realizamos uma revisão sistemática da literatura com o objetivo de verificar possíveis extensões existentes do diagrama de classes da UML para uso em SMAs, que, entre outras características importantes, suportem o modelo BDI.

A partir dos resultados desta revisão, identificamos lacunas nas atuais extensões do diagrama de classes da UML para o contexto de SMAs. Portanto, decidimos propor uma extensão para a linguagem de modelagem *Multi-Agent Systems Requirements Modeling Language* ou Linguagem de Modelagem de Requisitos de Sistemas Multiagentes (MASRML), adaptando e adicionando o diagrama de classes a ela, de tal forma que este diagrama possa suportar as necessidades identificadas na revisão sistemática.

Nossa extensão da Linguagem MASRML visa preencher as principais lacunas evidenciadas durante a nossa revisão sistemática de literatura, de tal forma que a linguagem MASRML seja capaz de representar estruturalmente, por meio de um diagrama de classes adaptado, os principais conceitos do modelo BDI. Após desenvolvermos uma proposta inicial da nossa extensão, avaliamos sua eficácia por meio de um experimento de comparação de modelos.

1.1 Motivação

Segundo Thung et al. (2014), um diagrama de classes aumenta nossa capacidade de compreender o projeto de software. Nessa linha, Guedes (2018) também afirma que o diagrama de Classes é um dos mais importantes e utilizados diagramas da UML, sendo que seu principal objetivo é permitir a visualização das classes que compõem o sistema, juntamente com seus respectivos atributos e métodos, além de demonstrar os relacionamentos entre essas classes.

Levando isso em consideração, percebemos a importância de aplicar esse diagrama em projetos de SMAs. No entanto, como o mencionado diagrama não possui recursos que permitam a modelagem de conceitos específicos de SMAs, torna-se necessário adaptar o diagrama de classes para o contexto desse tipo de software.

Desta maneira, com o objetivo de identificar as extensões da UML para SMAs, realizamos uma revisão sistemática da literatura buscando verificar como essas extensões suportam o uso do modelo BDI e a utilização de agentes como um todo.

Nessa revisão, analisamos se as atuais extensões do diagrama de classes para o contexto de SMAs auxiliam na representação desses conceitos e, com base nas lacunas encontradas, percebemos a necessidade de propor uma nova extensão da MASRML que possua mecanismos para superar os problemas identificados em outros trabalhos.

1.2 Objetivos da Pesquisa

Os objetivos específicos deste trabalho são:

- Realizar uma revisão sistemática da literatura para identificar possíveis extensões do Diagramas de Classe adaptadas ao contexto de SMAs;
- Aplicar, de forma prática, os trabalhos com as melhores pontuações, de acordo com a avaliação de qualidade dos trabalhos aceitos na revisão, analisando os pontos fortes e fracos dessas linguagens;
- Estender a MASRML por meio da adição de uma extensão do Diagrama de Classes adaptado para o contexto de SMAs;
- Avaliar a proposta, através de um Experimento com base em comparação de modelos.

1.3 Contribuições do Trabalho

As principais contribuições deste trabalho são:

- Apresentação de uma análise das extensões dos Diagramas de Classes da UML para o contexto de SMAs, por meio de uma revisão sistemática da literatura;
- Extensão da linguagem MASRML adicionando um diagrama de classes UML adaptado para permitir a representação de conceitos específicos de SMAs.

1.4 Organização do Trabalho

O restante do documento está organizado da seguinte maneira. O Capítulo 2 apresenta os conceitos necessários para o entendimento do trabalho como um todo. Neste

capítulo, são apresentados os conceitos essenciais para compreender um SMA e seus requisitos específicos. Também é abordado o processo de desenvolvimento de software ao qual o trabalho está relacionado, assim como a Linguagem MASRML utilizada. Além disso, são apresentados os conceitos necessários para compreender a extensão de diagrama de classes UML proposta. Em seguida, o Capítulo 3 apresenta a revisão sistemática de literatura realizada, em que foram analisadas as linguagens de modelagem baseadas da UML para o contexto de SMAs. O Capítulo 4, apresenta uma análise mais detalhada das linguagens que receberam uma melhor avaliação em nossa revisão sistemática de literatura. No Capítulo 4.6, é explicada a motivação por trás da proposta de uma extensão da linguagem MASRML para o contexto de SMAs e é apresentada a extensão do metamodelo da UML, com ênfase na descrição dos conceitos utilizados e em sua explicação. O Capítulo 5 detalha a avaliação realizada na extensão, descrevendo os passos realizados e os resultados obtidos com o experimento de comparação de modelos. E por último, o Capítulo 6 relata as considerações finais sobre a execução do trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

Esse capítulo apresenta os conceitos utilizados para desenvolver esse trabalho. Na Seção 2.1 são apresentados os conceitos de agente e Sistema Multiagente. O modelo BDI é apresentado na seção 2.2, detalhando sua estrutura e justificando seu uso. A Seção 2.2.1 apresenta as características de agentes suportadas pelo metamodelo apresentado nesse trabalho. Na Seção 2.4 é abordada a linguagem de modelagem de requisitos de Sistemas Multiagentes (MASRML). A Seção 2.5 apresenta uma explicação sobre processo *Requirements Engineering Process for MultiAgent Systems* ou Processo de Engenharia de Requisitos para Sistemas Multiagente (REPMAS). Na Seção 2.6 apresentamos uma descrição breve sobre o que é a linguagem de modelagem UML, juntamente com a subseção 2.6.1, que aborda o conceito de Diagrama de Classes UML. Por último, a Seção 2.7, que descreve o conceito de metamodelagem UML.

2.1 Sistemas Multiagentes

Um agente é um processo computacional estabelecido em um ambiente, projetado para atingir um propósito por meio de um comportamento autônomo e flexível (VICARI; GLUZ, 2007). Um agente é caracterizado por possuir autonomia, habilidade social, reatividade e proatividade. Autonomia é a capacidade do agente de operar sem a intervenção direta dos usuários. A habilidade social permite que os agentes interajam entre si por meio de alguma linguagem de comunicação própria dos agentes. A reatividade refere-se à capacidade de perceber o ambiente e responder oportunamente às mudanças que ocorrem nele. Por fim, a proatividade é a capacidade do agente de ter um comportamento direcionado a um objetivo, tomando a iniciativa sem receber ordens (HAJDUK; SUKOP; HAUN, 2019).

Um SMA é composto por vários agentes inteligentes interagindo entre si (WOOLDRIDGE, 2009). O desenvolvimento desse tipo de sistema é utilizado em diversas áreas (LIU L. ZHENG, 2011), devido, entre outras razões, à sua capacidade de lidar com a complexidade (BOES F. MIGEON, 2017).

2.2 Modelo Belief–Desire–Intention (BDI)

Segundo Rao, Georgeff et al. (1995), o modelo BDI é um modelo de software desenvolvido para programar agentes inteligentes, caracterizado por incluir crenças, desejos e intenções na arquitetura do agente. A seguir, detalharemos essas características.

- **Crenças:** representam o estado de informação que o agente possui, ou seja, suas crenças sobre o mundo (o ambiente em que se encontra), sobre si próprio e sobre outros agentes.

- **Desejos:** representam o estado motivacional do agente. Eles representam objetivos ou situações que o agente gostaria de alcançar.

- **Intenções:** representam os desejos que o agente crê que podem ser alcançados, de acordo com suas crenças e percepções. Quando um objetivo se torna uma intenção, o agente se compromete em atingi-lo e passa a agir para tal.

Na literatura, frequentemente os desejos são chamados de objetivos (*Goals*). Dessa forma, enquanto a crença de um agente não for verdadeira, ou seja, enquanto o agente não acreditar que um objetivo possa ser atingido, ele possui apenas o desejo de atingi-lo, mas não age para isso. A partir do momento em que a crença (ou crenças) tornar-se verdadeira, ou seja, o agente passar a acreditar que o objetivo possa ser atingido, esse objetivo passa a ser uma intenção e o agente passa a agir para atingi-lo, muitas vezes por meio da execução de um plano associado ao objetivo.

O modelo BDI é um modelo de software desenvolvido para programar agentes inteligentes caracterizado por incluir crenças, desejos e intenções na arquitetura do agente (BRATMAN et al., 1987). As crenças representam o que um agente acredita ser verdadeiro sobre o ambiente, sobre si mesmo e sobre outros agentes. Os desejos representam os objetivos que o agente gostaria de alcançar. Já as intenções representam desejos que o agente acredita que pode atingir e, portanto, age para alcançá-los (RAO; GEORGEFF, 1995).

Com relação à importância do modelo BDI para a área de AOSE, a seguir, apresentaremos alguns autores que defendem o uso desse modelo.

Getir, Challenger e Kardas (2014) argumentam que o modelo BDI é mais semelhante ao raciocínio humano e adequado para os agentes. Com relação às intenções dos agentes, Dennett (DENNETT, 1987) afirma que uma postura intencional trata as pessoas como agentes racionais que escolhem de acordo com suas crenças e desejos. Assim, é possível inferir quais crenças e desejos um agente deve ter para agir de forma racional. Além disso, Turner (TURNER, 2017) afirma que o princípio da intencionalidade capacita a previsão e explicação do comportamento humano com base em crenças e desejos.

Sobre o uso do modelo Singh (SINGH; PADGHAM; LOGAN, 2016) afirma que o modelo BDI tem sido amplamente utilizado no desenvolvimento de SMA. E Larsen (LARSEN, 2018) declara que esse modelo permite aos agentes um comportamento mais complexo do que os modelos reativos sem a sobrecarga computacional das arquiteturas cognitivas e que é mais fácil especificar o conhecimento. Finalizando, Alzetta (ALZETTA et al., 2020) afirma que este modelo representa uma das abordagens mais reconhecidas para integrar as habilidades cognitivas desejadas em agentes autônomos e facilita a descrição da relação causa-efeito para um agente atingir seus objetivos.

2.2.1 Características de Agentes de Software

2.3 Crenças dos Agentes

A definição de crença, conforme Silva, Meneguzzi e Logan (2020), é a representação das informações do agente sobre seu ambiente, sobre outros agentes e sobre si mesmo. Nessa mesma linha, Taillandier et al. (2017) afirma que as crenças são um conjunto de informações obtidas por meio de percepções ou comunicações entre agentes.

2.3.1 Objetivos dos Agentes (Goals)

Para Bonjean et al. (2014) *Goal* é um objetivo definido pelo projetista para um agente ou para todo o sistema. Nesta linha, Cossentino et al. (2014a) dizem que um *Goal* é uma descrição de um objetivo que o agente persegue e representa uma abstração de um estado de coisas que se deseja alcançar.

2.3.2 Percepções de Agente

Segundo Russell e Norvig (2016) a percepção 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.

2.3.3 Plano de Agente

Russell e Norvig (2016) definem um plano de agente como uma forma do agente atingir seus objetivos. Um plano é composto por ações que podem causar transições nos estados do ambiente.

Segundo García-Ojeda e Deloach (2014), um plano 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).

Cossentino e Seidita (2014) declaram que o comportamento de um agente é especificado em seu plano. O plano é a descrição de como combinar e ordenar tarefas e interações para cumprir total ou parcialmente um requisito.

2.3.4 Ação do Agente

Segundo Choren e Lucena (2005), uma ação é um ato computacional que resulta em uma alteração no estado do ambiente. Podem haver dois tipos de ação, sendo elas:

- **ação direta:** ação realizada enquanto o agente participa de um cenário para atingir um objetivo;
- **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.

2.3.5 Organização

Segundo Choren e Lucena (2005), uma organização é um grupo de (um ou mais) agentes trabalhando juntos para executar alguma função ou entregar um serviço. Em um SMA podem haver diversas organizações.

Cossentino et al. (2014b) definem uma organização como uma coleção de papéis que participam de padrões sistemáticos institucionalizados de interações com outros papéis em um contexto comum.

2.3.6 Papel de Agente

Cossentino e Seidita (2014) definem papel de agente como uma parte do comportamento social de um agente que é caracterizado por possuir um objetivo e/ou fornecer um serviço. O objetivo de cada papel é contribuir para o cumprimento de uma parte dos requisitos da organização em que ele está contido.

2.4 Linguagem MASRML

MASRML – é uma Linguagem de Modelagem Específica de Domínio (DSML - Domain Specific Modeling Language) baseada em UML, concebida para modelar requisitos de SMAs (GUEDES et al., 2020). Esta DSML estende o metamodelo UML para que o diagrama de casos de uso possa ser aplicado no domínio específico de SMAs. MASRML fornece mecanismos para representar os conceitos de papel do agente, objetivo (desejo), percepção, crença, intenção, plano e ação, suportando conceitos do modelo BDI.

Na MASRML, novos conceitos foram criados inspirados nos conceitos do padrão UML. A principal contribuição foi a criação das metaclasses AgentRoleActor e InternalUseCase para representar os papéis dos agentes e as funcionalidades atribuídas a eles, estereotipadas em objetivos, percepções, ações e planos. Algumas associações também foram criadas para associar percepções, ações e planos aos objetivos que um papel de agente deseja alcançar, estabelecendo assim as condições para que um objetivo se torne uma intenção e qual plano será executado neste caso, bem como as possíveis ações externas realizadas durante a execução de um plano.

Além disso, a documentação de casos de uso internos da MASRML permite, por exemplo, determinar o que é necessário para que um determinado objetivo se torne uma intenção ou estabelecer os passos a serem executados ao executar uma percepção ou um plano.

2.5 Processo REPMAS

O REPMAS (SOUZA; GUEDES; MENDONÇA, 2023) é um processo de Engenharia de Requisitos para SMAs. O processo engloba as quatro subáreas da engenharia de requisitos inspiradas no *Software Engineering Body of Knowledge* ou Guia para o Corpo de Conhecimento em Engenharia de Software (SWEBOK) (BOURQUE; FAIRLEY et al., 2014): elicitação, análise, especificação e validação. Este processo procura dar suporte ao engenheiro de software na hora de realizar a engenharia de requisitos para este tipo de sistema.

A etapa de elicitação desse processo visa descobrir requisitos específicos para SMA por meio de um contato direto com as partes interessadas (*Stakeholders*), elicitando agentes de software e suas funções, por meio de entrevistas, utilizando uma metáfora da organização que simula a contratação de novos funcionários para uma empresa. Essa elicitação tem como base a metodologia Homer (Wilmann; Sterling, 2005), metodologia criada para ser integrada em outros processos e que separa perguntas para o entrevistador realizar às partes interessadas.

A análise é realizada por meio da identificação de cenários, produzidos por meio da MASRML (GUEDES et al., 2020), linguagem que estende o diagrama de casos de uso UML para o contexto de SMAs. Esta análise visa identificar requisitos específicos para SMA com foco no suporte do modelo BDI. Por fim, a fase de validação desse processo visa validar os artefatos especificados nas fases de elicitação e análise.

2.6 UML

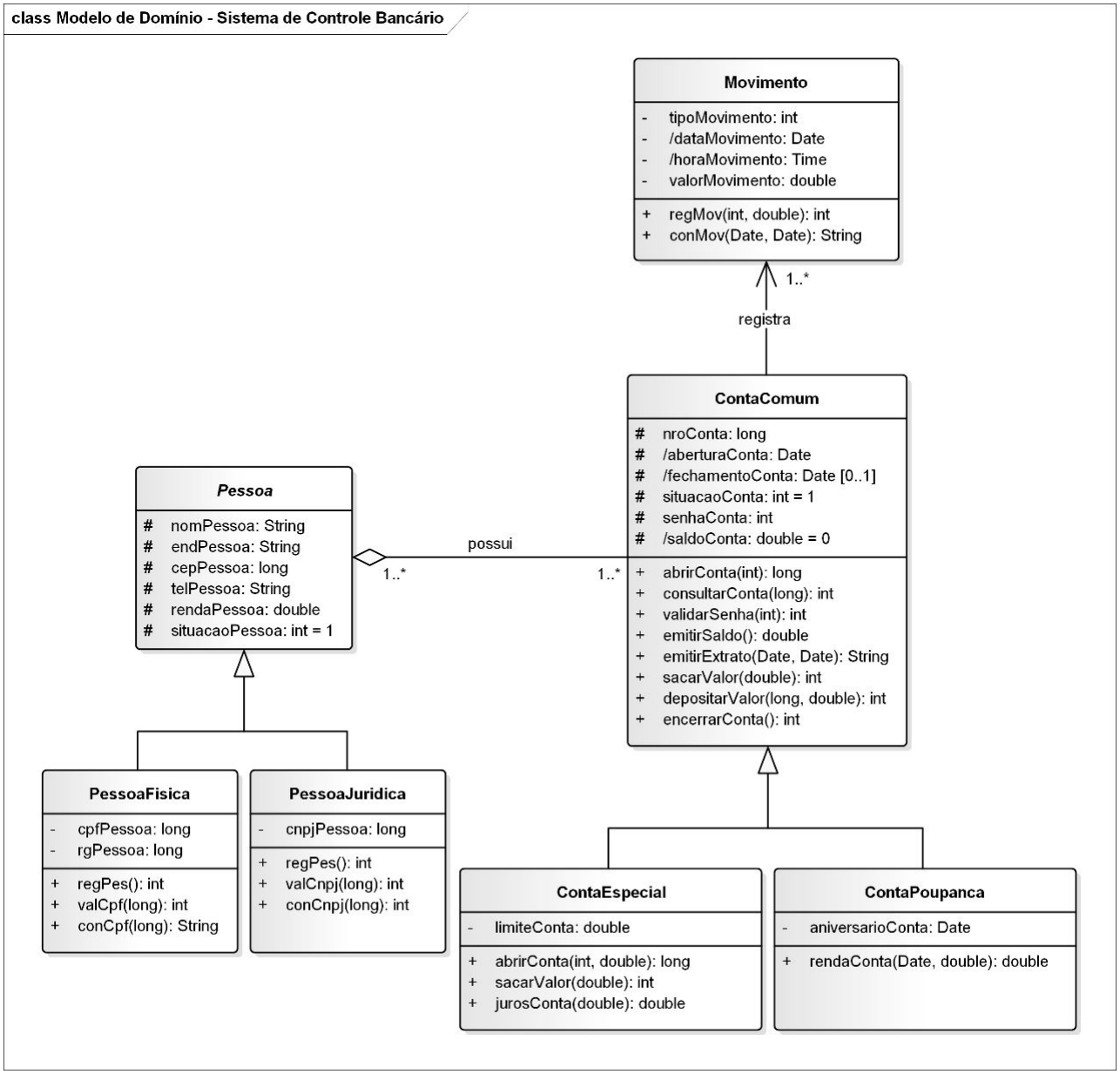
A UML é uma linguagem de modelagem de propósito geral que pode ser aplicada a todos os domínios de aplicação (GUEDES, 2018). Sendo a UML uma linguagem gráfica de modelagem, ela é utilizada para visualização, especificação, construção e documentação de artefatos de sistemas complexos de Software (BOOCH; RUMBAUGH; JACOBSON, 2006).

2.6.1 Diagrama de Classes UML

Os diagramas de classe UML são o núcleo principal da análise e projeto da Orientação a Objetos, a partir dos quais outros modelos são derivados (HERCHI; ABDESSALEM, 2012). Em se tratando do diagrama de classes, a OMG (OMG, 2011) não especifica quais elementos dos diagramas de classes UML devem ou não ser incluídos nos diagramas específicos e esta decisão é sempre deixada para os modeladores. Entretanto, a literatura sugere que os seguintes elementos do diagrama de classes UML são sugeridos como os mais importantes na modelagem conceitual (GREEN, 2000)(NITTO et al., 2002): Nome da classe, atributos de classes com tipos (tipos de dados primitivos ou estruturados),– associações entre as classes (incluindo agregação) com a multiplicidade especificada das

extremidades da associação e relações de generalização. Um exemplo de diagrama de classes, pode ser observado na Figura 1.

Figura 1 – Exemplo de diagrama de Classes.



Fonte: (GUEDES, 2018)

2.7 Metamodelo

Os metamodelos em ciência da computação e domínios relacionados são usados principalmente para facilitar a modelagem conceitual, definir construções das linguagens de modelagem conceituais, especificar restrições no uso de construções e codificar as semelhanças entre diferentes modelos (e metamodelos). Como a modelagem conceitual lida com a representação de conceitos, um elemento de um metamodelo também é um con-

ceito, ou um metaconceito, que é interpretado por todas as entidades definidas ou restritas pelo metaconceito (JEUSFELD, 2009).

Dentro da metamodelagem temos o uso de estereótipos, que segundo Gogolla e Henderson-Sellers (2002) são um mecanismo de extensão poderoso e potencialmente expressivo na *Unified Modeling Language* (UML). Segundo Guedes (2018) os estereótipos possibilitam certo grau de extensibilidade aos componentes ou associações da UML, além de permitir a identicação de componentes ou associações que, embora semelhantes aos outros, tenham alguma característica que os diferencie, dando-lhes mais destaque no diagrama. Nesse contexto, em Guedes (2018), podemos observar a existência de diversos estereótipos aplicados às classes do diagrama de classes, como os estereótipos "boundary", "control"e "entity", que servem para identificar, respectivamente, classes que representam classes de visão, relacionadas às interfaces do sistema com os usuários; classes de controle, responsáveis por interpretar eventos que ocorrem sobre as classes de visão; e classes de entidade, que estão diretamente relacionadas ao domínio do problema e contêm atributos e métodos para resolvê-lo.

3 REVISÃO SISTEMÁTICA SOBRE NOTAÇÕES QUE ESTENDEM O DIAGRAMA DE CLASSES DA UML PARA SMA

Este capítulo tem como objetivo apresentar os estudos que serviram como base para a análise da aplicação e/ou adaptação do diagrama de classes para o contexto de SMAs. Para selecionar esses estudos, foi realizada uma revisão sistemática da literatura com o objetivo de identificar as linguagens de modelagem derivadas da UML para o contexto de SMAs.

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

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

Dessa forma, a Seção 3.1 tem como objetivo apresentar o protocolo utilizado na RSL, juntamente com algumas decisões tomadas durante a revisão. A Seção 3.2 apresenta os resultados gerados durante a aplicação dos critérios de inclusão, exclusão e qualidade dos estudos. Por último, a Seção 3.3 apresenta os resultados obtidos com a execução das questões de extração.

3.1 Apresentação da Revisão Sistemática

A seguinte revisão foi realizada no período de 19/03/2022 à 09/12/2022 e teve como objetivo estabelecer primeiramente quais são as linguagens de modelagem derivadas da UML para o contexto de SMAs e, após a leitura dos artigos encontrados, determinar quais dessas linguagens adaptam o diagrama de classes e como foi realizada essa adaptação. Nosso principal interesse está em como essas linguagens representam as características do modelo BDI e a cobertura delas em relação aos pontos gerais de um SMA, como a comunicação entre os agentes e sua relação com o ambiente em que estão inseridos.

Tendo isso em vista, foi definido um protocolo de pesquisa para a revisão sistemática. Esse protocolo contém questões de pesquisa, critérios para identificar e selecionar estudos, critérios para inclusão ou exclusão de estudos, método de avaliação de qualidade, estratégia de extração de dados e também como conduzir a revisão.

3.1.1 Questões de Pesquisa

Para esta revisão sistemática, foram definidas cinco questões de pesquisa. A primeira questão de pesquisa (QP1) tem como objetivo identificar quais são as linguagens de modelagem baseadas na UML para SMAs.

A segunda questão de pesquisa (QP2) foi definida para identificar quais dessas linguagens adaptam o diagrama de Classes. Essa questão é a base para o nosso trabalho, fornecendo um panorama geral das linguagens que devemos analisar.

A terceira questão de pesquisa (QP3) tem como objetivo identificar como a comunicação entre agentes é representada nos diagramas de classes encontrados.

A quarta questão de pesquisa (QP4) visa identificar se os diagramas de classe apresentam características do ambiente em que os agentes estão inseridos.

E por último, a quinta questão de pesquisa (QP5) visa identificar como as características necessárias para o suporte ao modelo BDI são representadas.

3.1.2 Identificação e seleção de estudos primários

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

Para realizar esta revisão sistemática, foram utilizadas bases bibliográficas que atendem aos seguintes critérios: (I) possuem um mecanismo de busca baseado na web; (II) possuem um mecanismo capaz de utilizar palavras-chave; (III) contêm documentos da área de computação científica; e (IV) suas bases de dados são atualizadas regularmente.

Além disso, utilizamos como base para nossa revisão o capítulo intitulado *"Specific UML-Derived Languages for Modeling Multi-agent Systems"* (GUEDES; VICARI, 2022), que analisa Linguagens Específicas Derivadas de UML para Modelagem de SMAs. Esse trabalho nos auxiliou na identificação de palavras-chave mais adequadas para a *String* de busca. Durante a leitura desse capítulo, identificamos dois artigos que são citados e que não foram encontrados em nossas buscas nas bases bibliográficas. Portanto, esses artigos foram incluídos manualmente em nossa revisão.

A *String* genérica utilizada nesta revisão sistemática pode ser vista na Tabela 1.

Tabela 1 – *String* genérica.

String	Conector
(Language OR DSML OR "Domain Specific Modeling Language" OR notation)	AND
(UML OR "Unified Modeling Language")	AND
(Multiagent OR multi-agent OR Agent OR "Multi agent")	

Na Tabela 2, podem ser visualizadas as bases bibliográficas utilizadas e os critérios de filtragem aplicados.

Base de dados	**Filtro utilizado**
Scopus	Titulo, Resumo e Palavra chave
ACM	Titulo, Resumo e Palavra chave do Autor
Engineering Village	Titulo, Resumo e Palavra chave
IEEE	Sem filtros

Tabela 2 – Tabela de filtros utilizados

Além da busca utilizando a *String* genérica, foram aplicados filtros manuais nas bases bibliográficas, como apresentado na Tabela 2. Esses filtros manuais foram necessários devido ao alto número de resultados obtidos em algumas bases e à necessidade de excluir estudos que estavam fora do escopo da nossa pesquisa. Essa abordagem mais restrita foi adotada devido à natureza abrangente da nossa pesquisa, com o objetivo de encontrar o maior número possível de linguagens relevantes.

3.1.3 Critérios de inclusão e exclusão

O critério de seleção foi utilizado para identificar os estudos primários que forneceriam conteúdo relevante para responder às questões de pesquisa. Inicialmente, os estudos foram avaliados com base no título, resumo e palavras-chave. A revisão foi conduzida por três revisores, que analisaram os artigos obtidos nas bases de dados. Em caso de dúvidas sobre a classificação final de um estudo em relação aos critérios de inclusão ou exclusão, um especialista foi consultado. Os critérios de inclusão e exclusão estão detalhados na Tabela 3.

Tabela 3 – Critérios de inclusão e exclusão.

Crítério	**ID**	**Descrição**
Inclusão	IC1	O estudo deve apresentar uma notação para modelagem de requisitos de Sistemas Multiagentes baseada na notação UML.
Exclusão	EC1	Estudos que não apresentem uma notação para modelagem de requisitos de Sistemas Multiagentes baseada na notação UML;
	EC2	Estudos em qualquer idioma diferente do inglês;
	EC3	Estudos que não conseguimos acesso;
	EC4	Estudos disponíveis apenas na forma de resumos, apresentações de slides, pôsteres, apresentações introdutórias ou de aberturas sobre conferências ou artigos curtos;
	EC5	Estudos em que o foco é apresentar um sistema;
	EC6	Estudos que propõem uma notação para um tipo específico de aplicação;
	EC7	Estudos que não são artigos ou anais de eventos.

Fonte: O Autor

3.1.4 Avaliação de qualidade dos estudos

Definimos três critérios de qualidade para avaliar a relevância dos estudos em relação ao escopo desta pesquisa. Esses critérios foram utilizados para classificar os estudos mais relevantes, mas não para excluí-los. É importante ressaltar que esses critérios abrangem a notação como um todo, e não apenas a forma como ela adapta o diagrama de classes para o contexto de SMAs. Isso ocorre porque consideramos necessário analisar

notações que tenham adaptado outros modelos da UML e que possam ser estendidas para a adaptação do diagrama de classes. A seguir, descrevemos os critérios qualitativos e a pontuação atribuída a cada um deles.

- **CQ1**: Notação suporta o modelo BDI?
  - Sim (Y): Cobre totalmente os conceitos do modelo BDI;
  - Parcialmente (P): Contém as características do modelo, com limitações quanto ao funcionamento;
  - Não (N): Não cobre o modelo BDI.
- **CQ2**: O trabalho apresenta formas de como aplicar a notação?
  - Sim (Y): Apresenta como aplicar a notação;
  - Não (N): Não apresenta como aplicar.
- **CQ3**: O trabalho apresenta o metamodelo da notação?
  - Sim (Y): Apresenta o Metamodelo;
  - Não (N): Não apresenta o Metamodelo.

Para estabelecer um índice geral de qualidade dos estudos selecionados, atribuímos pontuações a cada critério definido. Utilizamos as seguintes pontuações: Sim (S) corresponde a 1, Parcialmente (P) corresponde a 0,5 e Não (N) corresponde a 0.

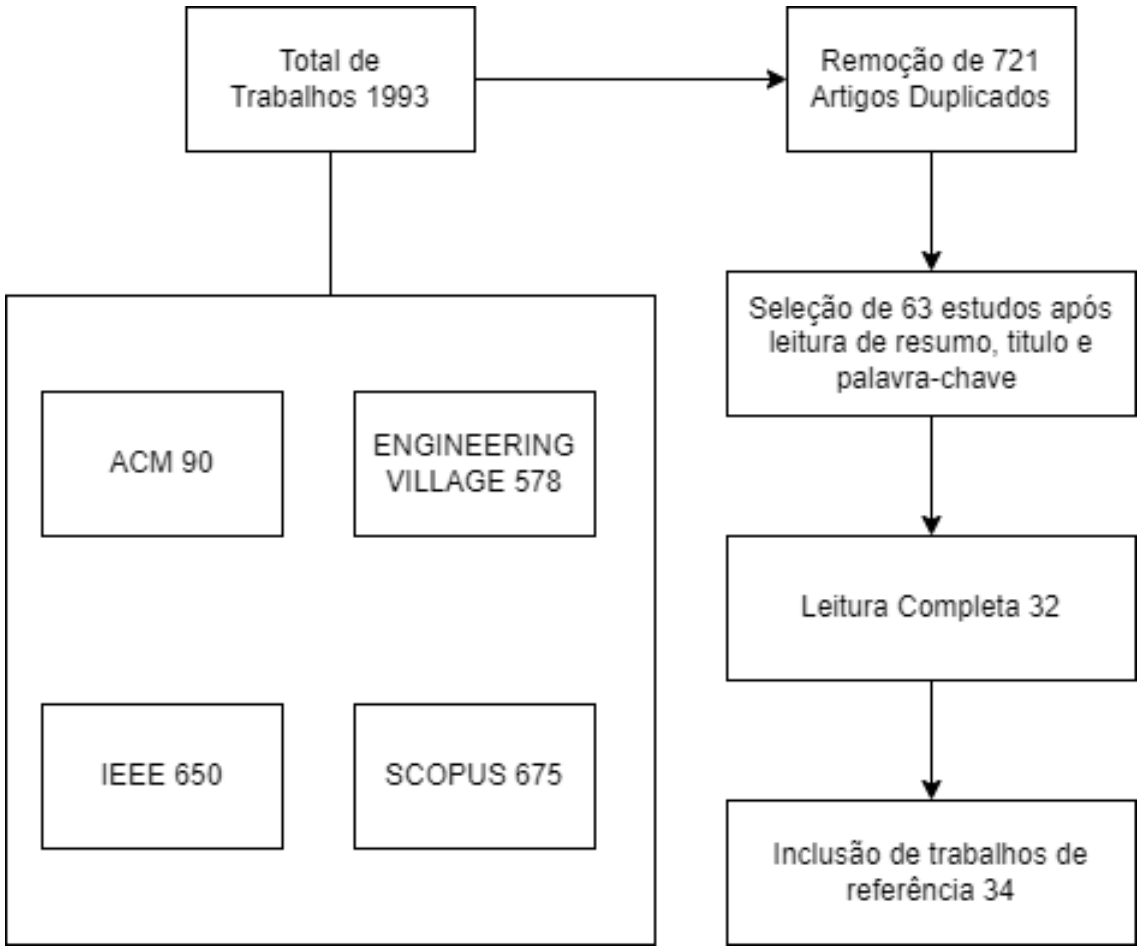
3.2 Etapa de Condução

A condução desta revisão sistemática foi realizada entre os meses de março e agosto de 2022. Foram definidas quatro etapas para a seleção dos estudos: (I) execução da *String* de busca nas bases bibliográficas; (II) remoção dos estudos duplicados; (III) aplicação dos critérios de inclusão e exclusão aos trabalhos; e (IV) leitura e extração das informações dos estudos resultantes da etapa.

No Estágio I, a pesquisa por *String* foi executada nas bases bibliográficas selecionadas para esta revisão. A visão geral desse estágio pode ser observada na Figura 2.

Essa etapa teve início com 1.993 trabalhos importados das bases bibliográficas selecionadas. No estágio II, um total de 721 estudos duplicados foram removidos. No estágio III, foram aplicados os critérios de inclusão e exclusão com base na leitura do título, resumo e palavras-chave, resultando na seleção de 63 estudos considerados promissores. Após a leitura completa dos trabalhos e inclusão manual de 2 artigos, chegamos a 34 trabalhos aceitos.

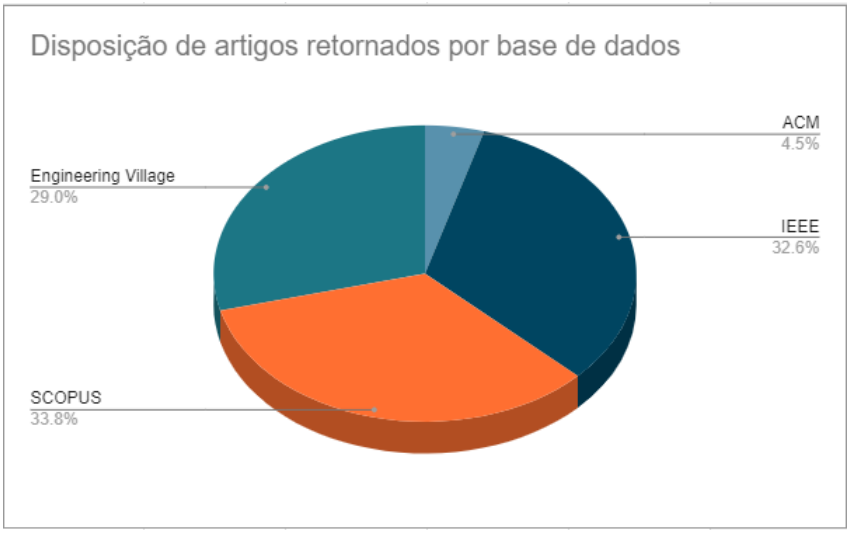
Figura 2 – Visão Geral dos estágios de condução da revisão.



Fonte: O Autor

Criamos dois gráficos para mostrar os resultados no estágio de condução. A Figura 3 apresenta a disposição de artigos retornados por cada base de dados e a Figura 4, apresenta um gráfico do resultado final de artigos aceitos, rejeitados e duplicados nesta revisão.

Figura 3 – Disposição de artigos retornados por base de dados



Fonte: O Autor

Figura 4 – Visão geral do estágio de condução.

Fonte: O Autor

3.2.1 Artigos aceitos e divisão das linguagens

Esta seção apresenta os artigos aceitos na revisão sistemática. Como nosso objetivo era analisar com maior precisão as linguagens, decidimos aceitar artigos que abordassem a mesma linguagem. Portanto, a Tabela 4 além de apresentar os artigos aceitos, também indica as linguagens abordadas por esses artigos.

Linguagem	**Referência(s)**
MAS-ML	(GONÇALVES et al., 2015), (GONÇALVES et al., 2010), (SILVA; CHOREN; LUCENA, 2006), (BADICA; BADICA, 2008), (SILVA; LUCENA, 2004) e (SILVA; NOYA; LUCENA, 2005)
AUML	(BAKAR; GHOUL, 2011), (SUBBURAJ; URBAN, 2018), (WINIKOFF, 2005) (HUGET; ODELL, 2004), (EHRLER; CRANEFIELD, 2004), (HUGET, 2004) (HUGET, 2002), (BAUER; MÜLLER; ODELL, 2001), (LAOUADI; MOKHATI; SERIDI-BOUCHELAGHEM, 2013), (BAUER; MULLER; ODELL, 2000), (BAUER, 2001), (LAOUADI; MOKHATI; SERIDI, 2014), (LAOUADI; MOKHATI; SERIDI-BOUCHELAGHEM, 2010), (BASTOS; RIBEIRO, 2004) e (BRESCIANI et al., 2004)
AML	(LIU et al., 2010), (TRENCANSKY; CERVENKA, 2005) e (TRENCANSKY; CERVENKA; GREENWOOD, 2006)
MESSAGE	(GARIJO; GOMEZ-SANZ; MASSONET, 2005) e (CAIRE et al., 2001)
AMOLA	(SPANOUDAKIS; MORAITIS, 2008)
ASM	(XIANG; ZHI-QING, 2009)
UML \Sysml	(ANTONOVA; BATCHKOVA, 2008)
MAM-UML	(BELLONI; MARCOS, 2004)
Soaml	(HAHN; JACOBI; RABER, 2010)
DAMRTS	(ADDOUCHE; ANTOINE; MONTMAIN, 2006)
AAM	(XIAO et al., 2007)
UML-AT	(FUENTES; GÓMEZ-SANZ; PAVÓN, 2006)
ADAM	(YIM; LEE; PARK, 2005)
UML para Tropos	(MYLOPOULOS; KOLP; CASTRO, 2001)
AOR	(WAGNER, 2003)
MA-UML	(HACHICHA; LOUKIL; GHEDIRA, 2009)
Massimo Cossentino notação	(COSSENTINO et al., 2012)
Notação de M. Dinsoreanu	(DINSOREANU; SALOMIE; PUSZTAI, 2002)
Notação de James P.Davis	(DAVIS; BONNELL, 2007)
Notação de Amelia Badica	(BADICA; BADICA, 2008)
JACK/UML	(PAPASIMEON; HEINZE, 2001)
Notação de Jan Murray	(MURRAY; STOLZENBURG, 2005)
Notação de Chiung-Hui Leon Lee	(LEE; LIU, 2002)
Notação de Carla Silva	(SILVA et al., 2006), (SILVA et al., 2007)
PASSI	(AZAIEZ; HUGET; OQUENDO, 2006)
Notação de Ralph Depke	(DEPKE; HAUSMANN; HECKEL, 2003)
Notação de Amir Zeid	(ZEID, 2002)
Notação de Piotr Kosiuczenko	(KOSIUCZENKO, 2002)
Notação de Federico Bergenti	(BERGENTI; POGGI, 2002)
Notação de Axel Wienberg	(WIENBERG; MATTHES; BOGER, 1999)
Notação de Manar Al-Kady	(AL-KADY; BAHGAT; FAHMY, 2008)
MASRML	(GUEDES et al., 2020)
SEA-ML	(GETIR; CHALLENGER; KARDAS, 2014)

Tabela 4 – Artigos Aceitos na Revisão

3.2.2 Qualidade das Linguagens selecionadas na Revisão

Para estabelecer um índice geral de qualidade dos estudos selecionados, atribuímos pontuações totais a cada critério definido. A Tabela 5 mostra a pontuação de cada estudo selecionado.

Tabela 5 – Índices de Qualidade dos Estudos.

ID	Estudo	CQ1	CQ2	CQ3	Total
S0	MAS-ML	P	S	S	2,50
S1	AUML	S	S	S	3.00
S2	AML	S	S	N	2.00
S3	MESSAGE	N	S	S	2.00
S4	AMOLA	N	S	N	1.00
S5	ASM	N	S	N	1.00
S6	UML/SysML	N	N	N	0.00
S7	MAM-UML	N	S	N	1.00
S8	Soaml	N	S	N	1.00
S9	DAMRTS	N	S	N	1.00
S10	AAM	N	N	N	0.00
S11	UML-AT	N	N	N	0.00
S12	ADAM	N	S	N	1.00
S13	UML Tropos	N	S	N	1.00
S14	AOR	S	S	N	2.00
S15	MA-UML	N	S	S	2.00
S16	Notação de Massimo Cossentino	S	S	N	2.00
S17	Notação de M. Dinsoreanu	N	S	N	1.00
S18	Notação de James P. Davis	N	S	S	2.00
S19	Notação de Amelia Badica	N	S	N	1.00
S20	JACK/UML	P	S	N	1.50
S21	Notação de Jan Murray	N	S	N	1.00
S22	Notação de Chiung-Hui Leon Lee	N	S	N	1.00
S23	Notação de Carla Silva	S	S	S	3.00
S24	PASSI	N	S	S	2.00
S25	Notação de Ralph Depke	N	S	N	1.00
S26	Notação de Amir Zeid	S	S	N	2.00
S27	Notação de Piotr Kosiuczenko	N	S	N	1.00
S28	Notação de Federico Bergenti	N	S	N	1.00
S29	Notação de Axel Wienberg	N	S	N	1.00
S31	Notação de Manar Al-Kady	P	N	S	1.50
S32	MASRML	S	S	S	3.00
S33	SEA-ML	S	S	S	3.00

Fonte: O Autor

Com a avaliação geral da qualidade dos trabalhos retornados, identificamos que 19 trabalhos foram considerados de baixa qualidade (pontuação de 0.1 a 1.0), 10 trabalhos foram considerados de qualidade mediana (pontuação de 1.1 a 2.0) e apenas 5 trabalhos receberam uma pontuação considerada boa (pontuação de 2.1 a 3.0).

Essa avaliação nos mostra que, apesar da existência de várias notações que adaptam a UML para o contexto de SMAs, a grande maioria dessas notações não atende aos nossos critérios de qualidade esperados para uma extensão. Tanto do ponto de vista conceitual, em relação ao suporte ao modelo BDI, quanto do ponto de vista prático, como a falta de exemplos de uso e a ausência de apresentação do metamodelo da extensão. Esses pontos dificultam o entendimento por parte daqueles que aplicam ou buscam aprimorar essas notações.

3.2.3 Etapa de Extração dos Dados

A extração de dados foi realizada nos trabalhos aceitos, utilizando uma estratégia que envolveu a definição de questões específicas a serem observadas nas linguagens selecionadas. Essas questões foram elaboradas para auxiliar os pesquisadores a responder às perguntas da pesquisa e são focadas exclusivamente na forma como as linguagens adaptam o diagrama de classes da UML para SMAs. As Questões de Extração (QE) definidas foram as seguintes:

- **QE01:** É possível identificar a interação dos agentes com o ambiente?
- **QE02:** É possível identificar a comunicação entre os agentes/papeis do agentes?
- **QE03:** É possível identificar as crenças iniciais de um agente ou de um papel do agente?
- **QE04:** É possível identificar as percepções associadas ao agente ou ao papel do agente?
- **QE05:** É possível identificar os objetivos associados ao agente ou ao papel do agente?
- **QE06:** A forma com que o modelo BDI é representado é coerente com a teoria do modelo?

3.3 Resultados da Revisão

Esta seção tem como objetivo apresentar os resultados obtidos com as respostas das questões de extração. É importante destacar que essas questões foram respondidas apenas para as linguagens que adaptam o diagrama de classes para o contexto de SMAs. Visto que as questões foram focadas em como a notação realizou a adaptação do diagrama de classes para o contexto de SMAs. A Tabela 6 apresenta os resultados obtidos com as respostas das questões de extração.

Linguagem	**QE01**	**QE02**	**QE03**	**QE04**	**QE05**
MAS-ML	P	P	S	S	S
AUML	N	P	S	N	N
MESSAGE	S	P	N	N	S
AMOLA	S	N	N	N	S
ADAM	N	S	N	N	N
UML para TROPOS	S	N	N	N	N
AOR	S	P	N	N	S
MA-UML	S	P	N	N	N
Notação para Jason	S	P	S	N	S
Notação de M. Dinsoreanu	S	N	N	N	N
Notação de James P. Davis	S	N	N	N	N
JACK/UML	S	N	S	N	N
Notação de Carla Silva	P	S	S	N	S
UML para PASSI	S	P	N	N	P
Notação de Ralph Depke	S	P	N	N	N
Notação de Amir Zeid	S	P	N	N	N
Notação de Hernan Mondani	N	P	N	N	S
Notação de Manar Al-Kady	S	N	N	N	N
SEA-ML	N	S	S	N	S

Tabela 6 – Resultado das Questões de Extração.

Com relação a Questão de Extração 06, o texto abaixo explica em detalhes o que identificamos como relevante no modo como as linguagens representam o modelo BDI. Destacamos que as linguagens MESSAGE, AMOLA, ADAM, UML para Tropos, MA-UML, Notação de Dinsoreanu, Notação de James P. Davis, UML para PASSI, Notação de Ralph Depke e Notação de Hernan Mondani, não suportam o uso do modelo BDI (como dito anteriormente em nossa questão de qualidade), portanto esta questão não se aplica a elas.

MAS-ML:

Consideramos parcial a coerência com que o modelo BDI é representado na linguagem MAS-ML. Apesar dos conceitos de objetivo e crença estarem presentes no modelo, o fato de ambas não estarem relacionadas torna impossível identificar quais crenças afetam o objetivo para que ele se torne uma intenção. Outro ponto importante é a falta de percepções, o que resulta na incapacidade dos agentes em perceber o ambiente e verificar

como as crenças são afetadas por ele. Em relação à comunicação com o ambiente, entendemos que a MAS-ML suporta uma interação parcial. Isso se deve ao fato de que, no diagrama de classes que apresenta os conceitos relacionados a informações internas dos agentes de software, a MAS-ML não suporta a interação com o ambiente. No entanto, a linguagem apresenta outro diagrama de classes que suporta esse recurso, mas não trabalha com conceitos internos de agentes (GUEDES; VICARI, 2022).

AUML: Consideramos que o diagrama de classes da AUML não suporta o uso do modelo BDI. Apesar da linguagem fornecer suporte, de certa forma, ao uso do modelo, esse suporte se limita ao diagrama de sequência, onde é possível descrever, por meio de pré e pós-condições, como uma crença afeta um objetivo e como um objetivo se torna uma intenção. No entanto, o diagrama de classes não nos permite representar objetivos, o que torna impossível a representação do modelo BDI.

AOR: Apesar da linguagem suportar o uso do modelo BDI nos diagramas mentais, o diagrama de classes da linguagem não oferece suporte para a representação do modelo BDI.

Notação para Jason: Esta notação apresenta um diagrama focado inteiramente na estrutura da linguagem Jason, uma linguagem para desenvolvimento de SMAs. Deste modo, é importante destacar que o suporte para o modelo BDI no diagrama segue as mesmas limitações presentes na linguagem em si. Identificamos algumas inconsistências nesse suporte, como o fato dos planos modelados receberem um excesso de responsabilidade, com a maioria das crenças estando relacionadas a eles e a possibilidade de existir planos sem estarem relacionados a um objetivo específico. Outra limitação importante é a ausência das percepções dos agentes. Além disso, em casos em que existem dois ou mais objetivos, não é possível verificar quais planos se referem a cada objetivo.

Jack/UML: Esta notação apresenta uma forma de modelar agentes BDI com base na estrutura presente na linguagem de programação JACK. Nessa notação, um agente é a entidade principal que atua como um módulo autônomo e possui planos (sequência de ações), base de dados (crenças dos agentes) e eventos (respostas dos agentes). No entanto, essa forma de adaptação do modelo apresenta algumas limitações conceituais. A mais significativa é a ausência de objetivos dos agentes, o que levanta questionamentos sobre a validade desse suporte. Além disso, os planos dos agentes assumem funções que originalmente não seriam suas, como o gerenciamento das crenças dos agentes.

Notação de Carla Silva: Esta notação apresenta os conceitos relacionados ao modelo BDI. No diagrama de classes, cada crença, plano e objetivo do agente é representado por uma classe distinta. No entanto, identificamos problemas conceituais na notação. Por exemplo, as crenças não podem estar diretamente relacionadas aos objetivos do agente, tornando impossível identificar como uma crença afeta um objetivo específico. Isso levanta dúvidas sobre até que ponto a notação realmente suporta o modelo BDI. Além disso, observamos que a notação não oferece suporte aos conceitos de percepção e

ação, que são elementos importantes na execução do modelo BDI.

Notação de Amir Zeid: O artigo menciona que a notação suporta o uso do modelo BDI, no entanto, não foi possível identificar como esse suporte é realizado. Os diagramas apresentados na notação não utilizam o modelo BDI e o texto não oferece informações claras sobre como aplicar o modelo nessa notação.

Notação de Manar Al-Kady: O artigo que descreve a notação é um estudo primário que não fornece detalhes suficientes sobre o funcionamento da mesma. Os autores mencionam que a intenção do artigo é apresentar uma extensão simples e genérica do metamodelo UML para representar SMAs. Portanto, podemos identificar apenas os conceitos presentes em seu metamodelo, como o uso de crenças e objetivos, mas não é possível obter informações detalhadas sobre seu funcionamento devido ao fato de que a sintaxe é muito abstrata. Além disso, é importante ressaltar que a linguagem não utiliza conceitos fundamentais, como percepções, ações e planos.

SEA-ML: A notação SEA-ML apresenta os conceitos de crença e objetivo. No entanto, consideramos seu suporte ao modelo BDI como parcial. A falta de uma relação clara entre os objetivos, as crenças e os planos torna impossível determinar quais crenças afetam um objetivo específico e quais planos são acionados para cada objetivo. Além disso, a notação não aborda a percepção no diagrama de classes, o que impede a representação de como o agente percebe o ambiente.

4 ANÁLISE DOS DIAGRAMAS DE CLASSE PRODUZIDOS APLICANDO AS LINGUAGENS/NOTAÇÕES APROVADAS PELOS CRITÉRIOS DE QUALIDADE

Neste capítulo, detalhamos a aplicação de quatro notações que foram consideradas "boas" de acordo com os critérios de qualidade da revisão sistemática. Nosso objetivo foi compreender melhor o funcionamento dessas notações, identificando seus pontos fortes e fracos. Através dessa análise, buscamos identificar as lacunas existentes que justificaram a necessidade de estender a linguagem MASRML para oferecer suporte ao uso do diagrama de classes no contexto de SMAs.

Ressaltamos que, diferentemente da análise realizada no capítulo que detalha os resultados da revisão sistemática, este capítulo contém uma análise mais aprofundada das notações, realizada após aplicá-las de forma prática.

Para aplicar as notações, utilizamos como base um SMA para edição de objetos de aprendizagem (AGUIAR; FLÔRES, 2014). A descrição do sistema é apresentada por Guedes em (GUEDES, 2012). A seguir, iremos detalhar o funcionamento desse sistema.

O principal objetivo do sistema é dar apoio à edição e criação de objetos de aprendizagem. Desta forma, um Objeto de Aprendizagem (OA) pode ser visto como um artefato composto de duas camadas (ou níveis):

- Camada dos Metadados: que engloba as informações de catálogo do OA, descrevendo sua composição e outras informações relevantes. Entre essas informações, destacamos os dados descritivos utilizados para busca, localização, recuperação e apresentação do conteúdo.

- Camada de Conteúdo: que contém o material de aprendizado em si, conforme descrito pelos metadados. Essa camada é visualizada pelo usuário com o objetivo de alcançar os objetivos específicos de uma determinada lição.

Sabendo que os metadados são informações descritivas dos objetos de aprendizagem, é importante que qualquer alteração no formato do conteúdo ou a existência de diferentes formatos para o mesmo conteúdo sejam devidamente registradas nos metadados. Isso garante a unificação do catálogo em relação a um OA, mesmo que o objeto possa oferecer visualizações distintas.

Portanto, para possibilitar a reutilização de um OA em várias plataformas, é necessário adaptar tanto os metadados quanto o conteúdo do OA. Isso garante a consistência das informações e a adequação às diferentes plataformas de visualização.

A principal funcionalidade prevista para o SMA de Autoria de OA é a criação e alteração de conteúdos desses objetos. A edição de conteúdo permite transformar o objeto em um formato descrito e aceito por uma infraestrutura específica para um novo OA, seguindo as instruções do desenvolvedor de OA ou criados por um agente. É importante

destacar que as infraestruturas que recebem o OA traduzido não fazem parte do sistema que iremos modelar, uma vez que nosso foco está exclusivamente na lógica interna do agente.

4.1 Aplicação da Notação MAS-ML

Segundo Silva (SILVA; LUCENA, 2003) a MAS-ML (SILVA; LUCENA, 2004) (SILVA; NOYA; LUCENA, 2005) (SILVA; CHOREN; LUCENA, 2006) (BADICA; BADICA, 2008) (GONÇALVES et al., 2010) (GONÇALVES et al., 2015) foi construída sobre a superestrutura da UML 2.1, estendendo-a por meio de diversos novos conceitos de modelagem representados por uma nova metaclasse e estereótipos. Essa extensão foi considerada adequada para capturar as características típicas de SMAs.

A Figura 5 apresenta nossa aplicação do diagrama Organizacional da MAS-ML. Além deste diagrama, a MAS-ML também suporta outros dois diagramas baseados no diagrama de classes da UML: o diagrama de classes da MAS-ML e o diagrama de papel.

Entretanto, consideramos que apenas o diagrama organizacional já contribuiria com nossa análise, entre outros motivos, porque o diagrama de classes nos parece conceitualmente incorreto. Os autores da MAS-ML chamam de ”objetos” elementos que são claramente classes. Além disso, o diagrama de classes não deve conter objetos, e sim, como seu nome diz, apenas classes. Na UML, existe um diagrama específico para representar objetos, chamado justamente diagrama de objetos (GUEDES, 2009). Além disso, na UML objetos não possuem métodos, apenas atributos. Os exemplos apresentados na MAS-ML misturam classes de organização, classes de agentes e classes normais. As classes de organização não acrescentam muita informação ao diagrama e são uma informação à parte. As classes de agente são as mesmas do diagrama organizacional e, no diagrama de classes, apenas demonstram as classes normais que os agentes poderiam acessar.

O diagrama de papéis de objeto da MAS-ML não nos parece muito útil, pelo mesmo motivo dito anteriormente, uma vez que os elementos chamados de ”objetos” e ”papéis de objeto” são conceitualmente classes, visto que possuem métodos, o que não é correto em objetos na UML. Além disso, os papéis de objeto servem apenas para demonstrar os tipos de objeto que podem assumir os papéis em questão, semelhante aos papéis de agente assumidos por agentes. Os exemplos apresentados na MAS-ML representam os papéis de objeto ”avaliação geral” ou ”revisão cega” assumidos pelo objeto ”revisão”. Outro exemplo são os papéis de objeto ”submissão” e ’camera ready” assumidos pelo objeto ”artigo”. Acreditamos que esse tipo de informação poderia ser facilmente representado por mecanismos da própria UML, como a criação de um novo atributo para identificar o tipo de objeto.

O diagrama de papel também demonstra quais papéis de objeto podem ser assumidos por quais papéis de agente. Entretanto, a característica mais importante do diagrama de papéis é demonstrar as associações entre os papéis de agente. Ele permite verificar se

algum papel foi derivado de outro ou se um papel de agente é superior ou subordinado a outro. Porém, na nossa opinião, em relação ao suporte ao modelo BDI, não acrescenta informações relevantes.

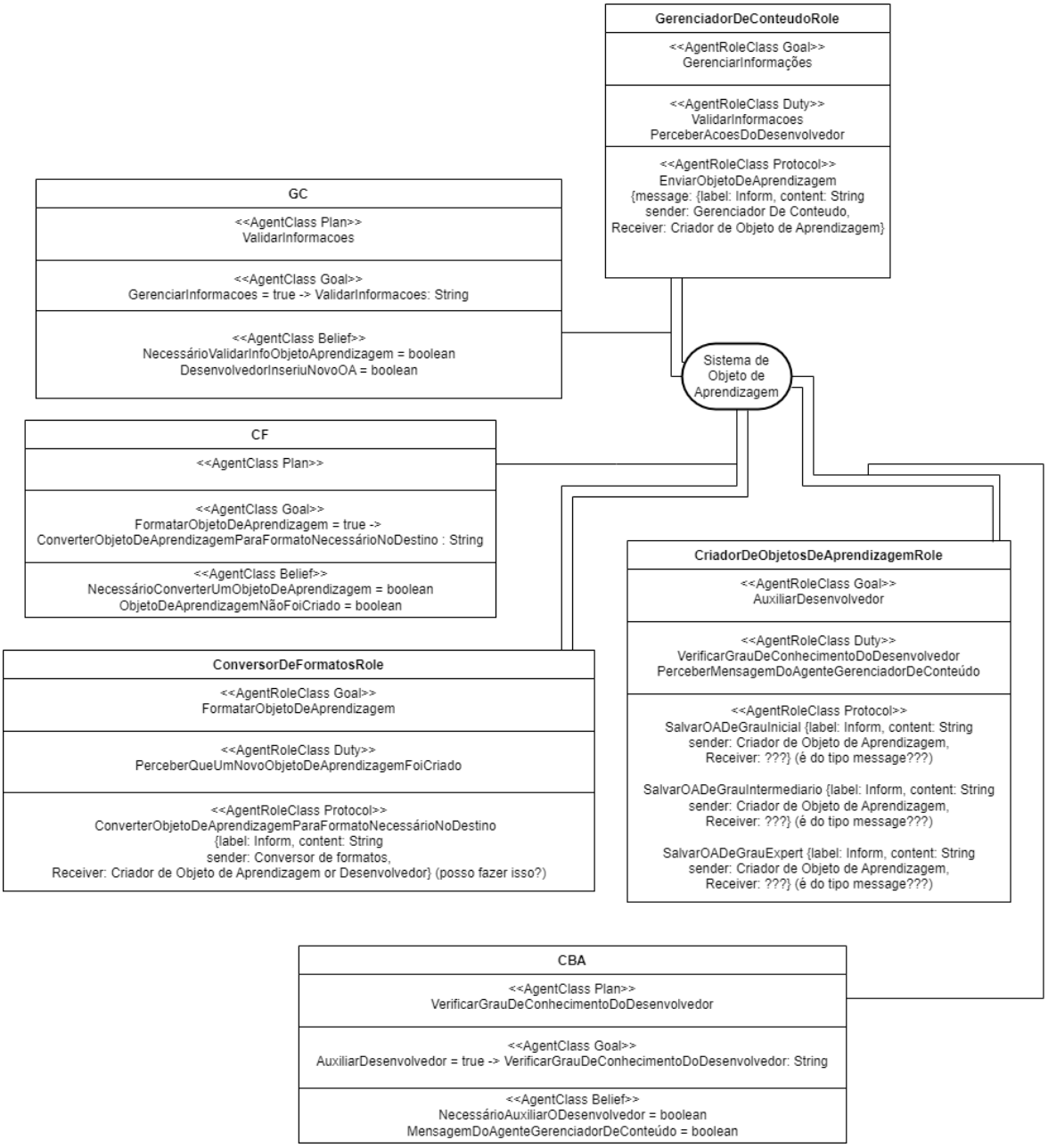
O diagrama organizacional da MAS-ML permite modelar agentes, papéis de agentes, objetivos, planos, crenças, deveres e protocolos de comunicação. No entanto, alguns conceitos não possuem uma notação específica, como objetivos, crenças, deveres e direitos, que são modelados dentro das classes Agent ou AgentRole e representados como atributos estereotipados, não possuindo um símbolo próprio. Consideramos que essa estratégia pode se tornar um problema caso o número de crenças, objetivos e outros atributos seja grande, pois isso pode tornar as classes modeladas extensas.

Na MAS-ML, não encontramos uma forma explícita de modelar percepções, o que torna pouco claro como o agente percebe o ambiente. Na nossa modelagem, consideramos as percepções como deveres (Duty), porém, essa abordagem não parece ser correta. A falta de representação das percepções também impede a identificação dos objetivos e crenças que são afetados por mudanças no ambiente, causadas por usuários ou outros agentes, por exemplo. Essa lacuna na notação dificulta a compreensão do processo de percepção e sua relação com os demais elementos do SMA.

No entanto, o capítulo que utilizamos como base para nossa revisão (GUEDES; VICARI, 2022) menciona que a MAS-ML possui uma evolução, a MAS-ML 2.0, que adiciona a metaclasse AgentPerceptionFunction, permitindo a modelagem de percepções. No entanto, é importante ressaltar que a versão padrão da MAS-ML é a de Silva (SILVA; CHOREN; LUCENA, 2008). Portanto, considerando a versão padrão da notação, a modelagem de percepções não é adequadamente abordada.

Outro ponto negativo que podemos destacar na MAS-ML é a representação dos planos na classe de agente, em vez de na classe de papel do agente. Isso pode ser considerado problemático, uma vez que espera-se que um plano seja acionado quando um agente está desempenhando um papel no sistema, ou seja, o plano deveria ser algo que o papel executa. Poderíamos considerar que um agente poderia ter objetivos e crenças iniciais antes de assumir um papel, mas uma vez assumido o papel, os objetivos e crenças associados a ele teriam prioridade sobre os do agente. Dessa forma, é bastante estranho que os papéis não tenham planos, enquanto os agentes os tenham. Essa representação dá a impressão de que os objetivos se repetem tanto no agente quanto no plano, resultando em redundância.

Figura 5 – Diagrama Organizacional da MAS-ML.



Fonte: O Autor

4.2 Aplicação da Notação AUML

A linguagem AUML (BAUER; MULLER; ODELL, 2000) (BAUER; MÜLLER; ODELL, 2001) (BAUER, 2001) (HUGET, 2002) (BASTOS; RIBEIRO, 2004) (HUGET; ODELL, 2004) (EHRLER; CRANEFIELD, 2004) (HUGET, 2004) (BRESCIANI et al., 2004) (WINIKOFF, 2005) (LAOUADI; MOKHATI; SERIDI-BOUCHELAGHEM, 2010) (BAKAR; GHOUL, 2011) (LAOUADI; MOKHATI; SERIDI-BOUCHELAGHEM, 2013) (LAOUADI; MOKHATI; SERIDI, 2014) (SUBBURAJ; URBAN, 2018)
foi uma das primeiras linguagens derivadas da UML especificamente para modelar SMAs. Seu uso foi proposto em algumas metodologias/processos AOSE como Gaia (CERNUZZI; ZAMBONELLI et al., 2004) (CERNUZZI et al., 2004), Prometheus (PADGHAM; WINIKOFF, 2002) ou Roadmap (JUAN; PEARCE; STERLING, 2002).

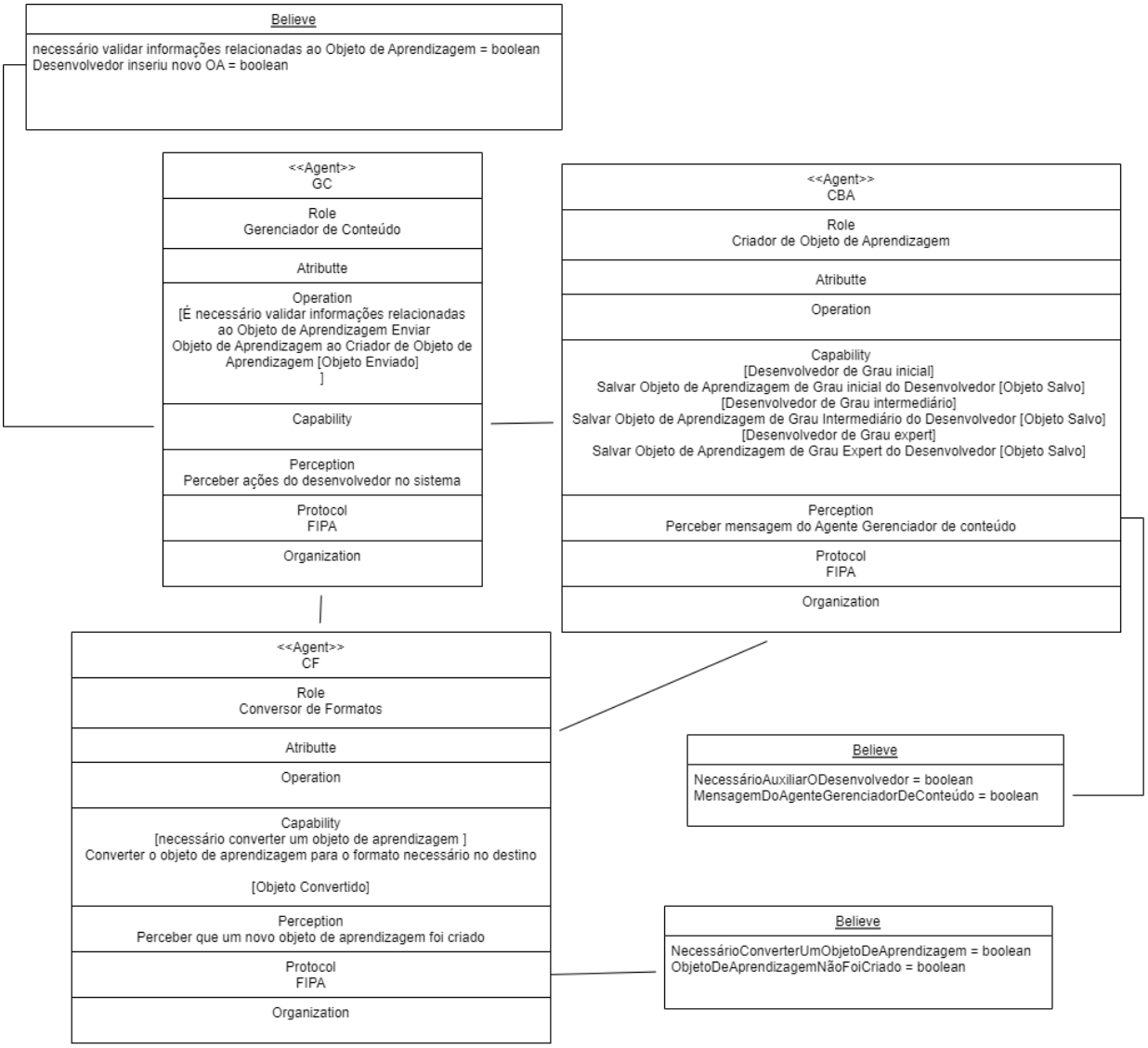
Como mencionado anteriormente, o suporte ao modelo BDI da AUML se limita ao diagrama de sequência, e mesmo assim de forma limitada. No entanto, a ideia inicial de realizar a análise de qualidade para toda a notação, e não apenas focada no diagrama de classes, é verificar a possibilidade de estender a linguagem e melhorar os aspectos necessários.

Desta forma, a Figura 6 apresenta a nossa aplicação da linguagem AUML. A AUML abrange os conceitos de agente, papel de agente, atributos, capacidades, percepções, protocolo, organização e crenças. No entanto, consideramos que esses conceitos não detalham suficientemente um agente BDI, sendo necessário acrescentar o conceito de objetivos do agente.

Entretanto, a linguagem AUML também apresenta a mesma limitação que observamos na MAS-ML, que é a falta de classes próprias para representar as características específicas do agente, resultando em classes complexas. Além disso, mesmo a classe "believe" (crenças), que é representada como uma classe separada do agente, não possui um propósito visível no diagrama, tornando difícil entender como as crenças afetam as características do agente. O mesmo ocorre com as percepções.

Outro ponto relevante é que, ao contrário da MAS-ML, a AUML não define uma forma padrão de modelar o protocolo de comunicação do agente, o que torna sua utilidade pouco clara.

Figura 6 – Diagrama de classes da notação AUML.



Fonte: O Autor

4.3 Aplicação da Notação de Carla Silva

A notação proposta por Carla Silva (SILVA et al., 2006) (SILVA et al., 2007) foi desenvolvida com o objetivo de facilitar o desenvolvimento eficiente de sistemas baseados em agentes. Essa linguagem propõe uma extensão do metamodelo UML 2.0 para incorporar características específicas de agentes de software, permitindo a criação de diagramas baseados em UML que capturam quatro visões estruturais de SMAs: Arquitetura, Intenção, Ambiente e Comunicação.

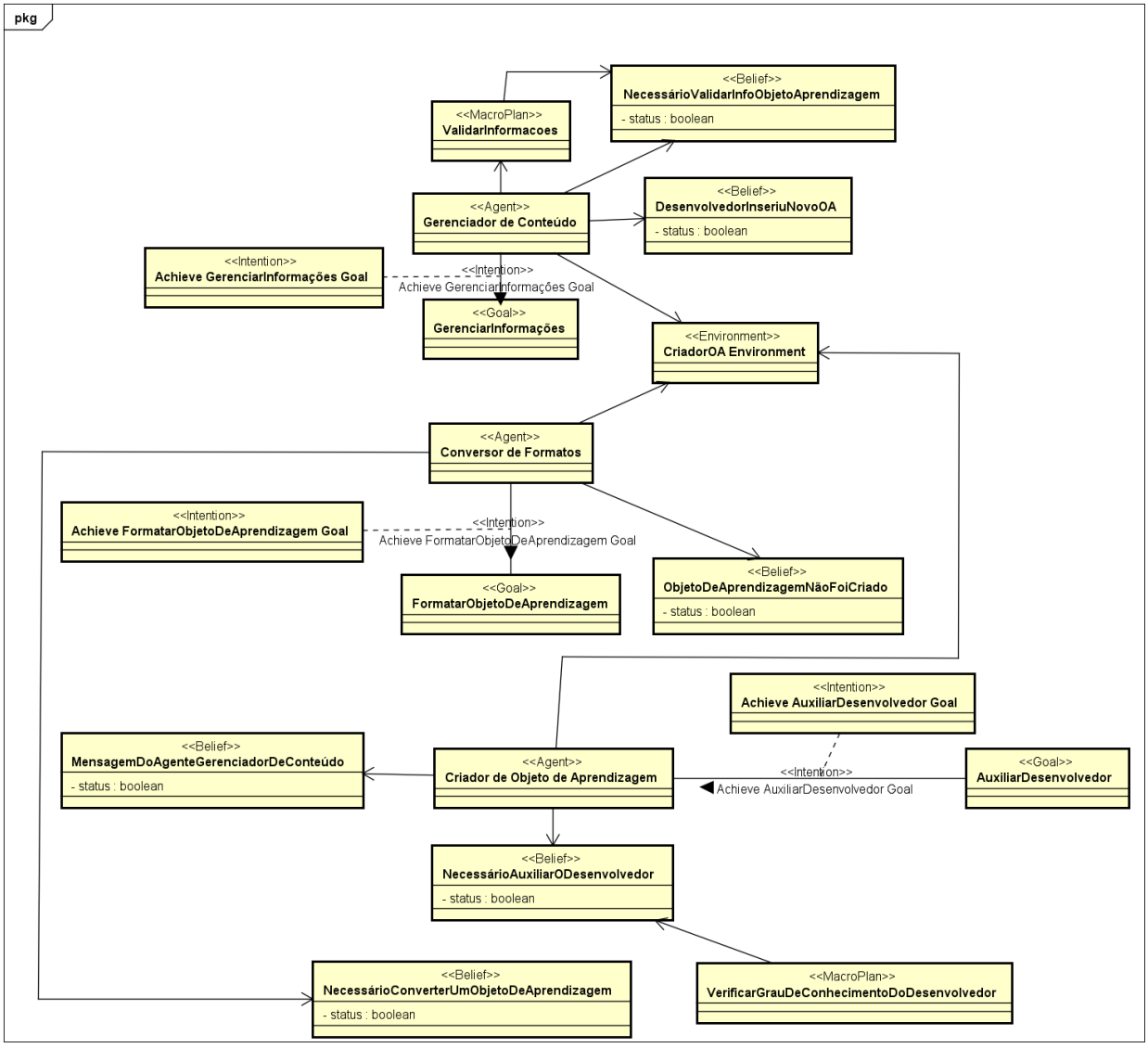
A aplicação da notação pode ser observada na Figura 7, que apresenta os conceitos de agente, crença, objetivo, macro plano, micro plano e ambiente. Conforme mencionado na Seção 3.3, cada característica do agente é representada por uma nova classe, o que

consideramos adequado para sistemas complexos. No entanto, essas novas classes contêm apenas uma descrição, não possuindo complexidade suficiente para justificar a criação de uma classe separada. Como resultado, o diagrama apresenta uma série de conexões 1 para 1, o que pode dificultar a compreensão visual e gerar poluição.

Além desse ponto, consideramos que a classe de ambiente (*environment*) deveria conter atributos que contribuíssem para a integração do agente com o ambiente em que ele está inserido. Da mesma forma, seria necessário incluir as ações do agente que podem estar relacionadas a essa classe de ambiente.

Outro ponto relevante a ser mencionado é a falta do conceito de percepção. Essa notação não apresenta uma forma de identificar como o agente percebe o ambiente, o que dificulta o entendimento do funcionamento do sistema.

Figura 7 – Digrama de classes da notação de Carla Silva.



Fonte: O Autor

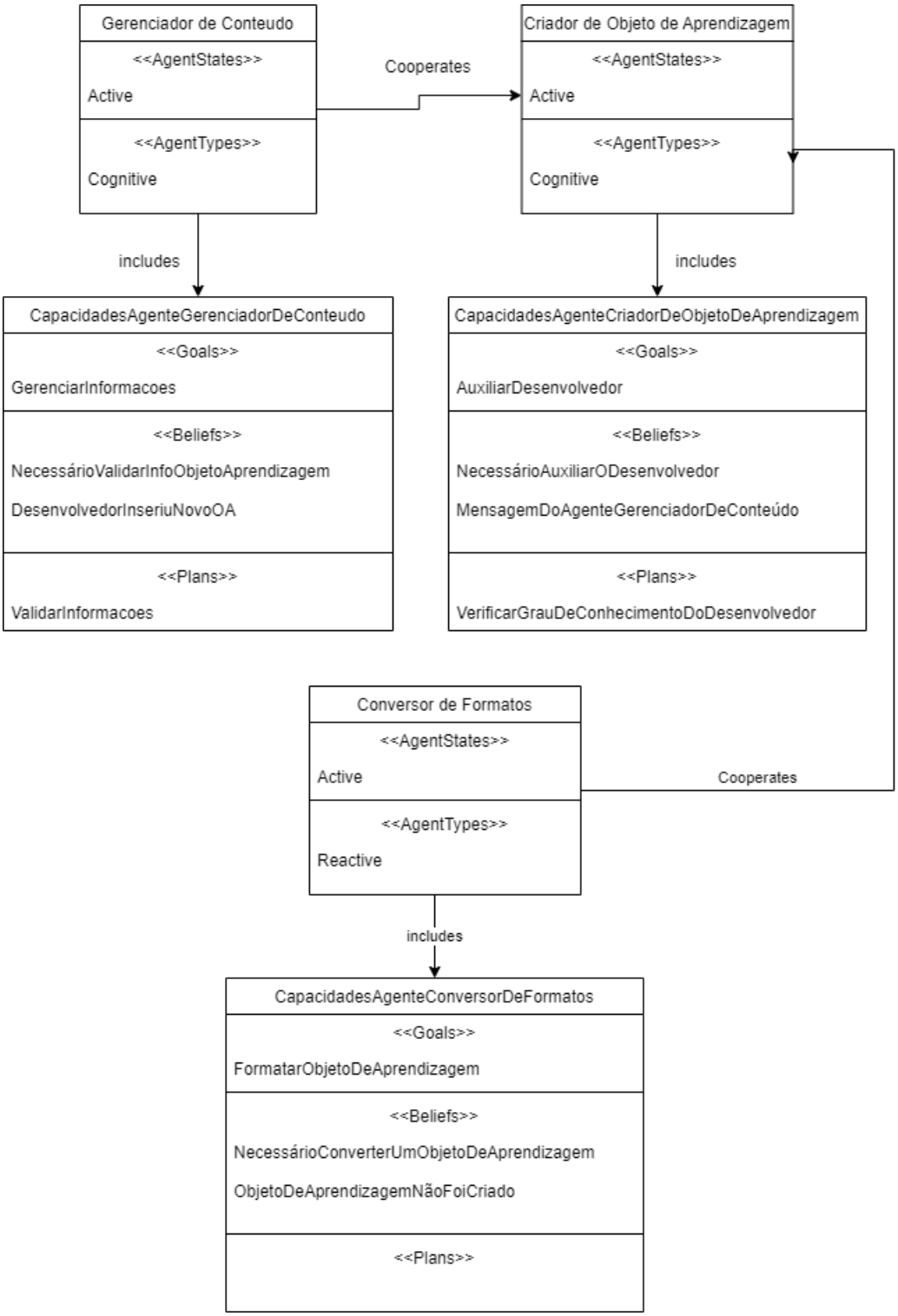
4.4 Aplicação da Notação SEA-ML

O metamodelo SEA-ML (GETIR; CHALLENGER; KARDAS, 2014) oferece suporte a muitos recursos dos agentes, incluindo o modelo BDI. No entanto, tivemos dificuldade em aplicar a notação devido ao fato de que a linguagem é específica para o uso de agentes em Web Semântica. Acreditamos que a linguagem poderia ser mais genérica para abranger uma gama mais ampla de tipos de SMA. Outro ponto importante é que a linguagem é estruturada em vários "pontos de vista", sendo dividida em diferentes diagramas, dependendo da função. Como nosso objetivo principal é verificar o funcionamento do modelo BDI, limitamo-nos a aplicar o diagrama de visão interna do agente, pois entendemos que este é o que mostra as características do agente

A Figura 8 apresenta a aplicação que realizamos na notação SEA-ML. Um ponto importante a ser observado é que o diagrama de classes não apresenta suporte para o uso de papéis de agente. Diante disso, optamos por utilizar a modelagem do agente para representar o papel de agente. No entanto, a notação SEA-ML oferece suporte ao uso de papéis, o que nos deixou questionando por que o papel de agente não é utilizado neste diagrama.

Com relação ao ponto de vista interno do agente, é válido observar que, embora o metamodelo esteja bem estruturado, a classe de capacidade, que engloba crenças, objetivos e planos, pode gerar uma sobrecarga de informações em um único componente. Isso pode resultar em um diagrama desnecessariamente complexo, especialmente quando um agente possui um grande número de crenças, objetivos ou planos.

Figura 8 – Digrama de classes da notação SEA-ML.



Fonte: O Autor

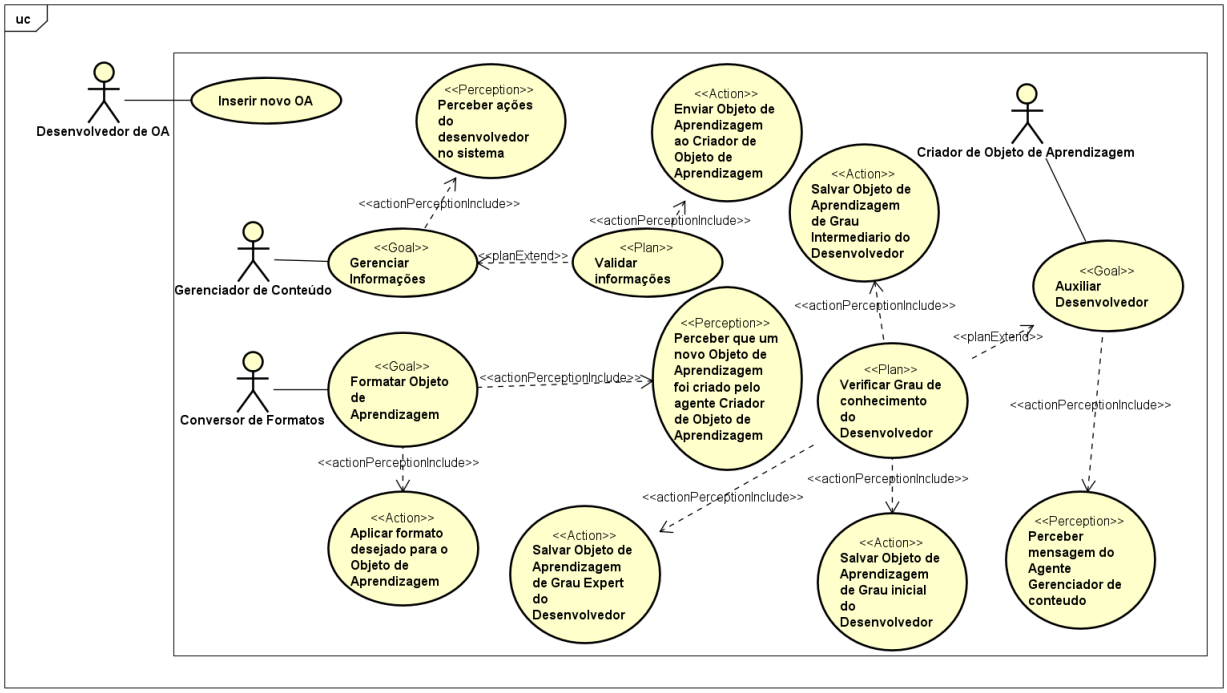
4.5 Aplicação da Notação MASRML

A MASRML foi considerada uma notação de qualidade em nossa revisão devido às características que ela apresenta. Ela oferece suporte ao modelo BDI, o que é muito importante para a modelagem de sistemas inteligentes. Além disso, apresenta formas claras de aplicação da notação, facilitando a compreensão e utilização da linguagem. Por último, a MASRML apresenta seu metamodelo no artigo base, o que permite uma compreensão mais aprofundada dos conceitos da notação e de suas possibilidades.

Todavia, a MASRML não suporta o diagrama de classes. Por isso, foi necessário avaliar a possibilidade de estender esta linguagem de maneira a incluir conceitos estendidos a partir do diagrama de classes da UML. Para isso, foi aplicado o diagrama de casos de uso da MASRML na modelagem do sistema de Criação de Objetos de Aprendizagem, com o objetivo de verificar se a adição de uma extensão do diagrama de classes à linguagem MASRML poderia ser vantajoso na modelagem de SMAs.

A Figura 9 apresenta o diagrama de casos de uso da MASRML. Podemos verificar que a linguagem MASRML descreve como os agentes percebem o ambiente por meio de suas percepções, e quais objetivos do agente são afetados por essas percepções. Dessa forma, também podemos identificar quais planos são disparados pelos objetivos e quais ações esses planos irão realizar para afetar o ambiente.

Figura 9 – Digrama de casos de uso da MASRML.



Fonte: O Autor

Todos esses pontos representam lacunas que identificamos nas outras notações e consideramos necessários para adaptar o modelo BDI. No entanto, a linguagem MASRML também apresenta uma deficiência em sua representação gráfica, em que as crenças dos agentes só podem ser visualizadas por meio da descrição textual de cada caso de uso, como pode ser observado na Tabela 7 no caso de uso interno "Gerenciar Informações".

Tabela 7 – *Goal* - Gerenciar Informações.

Internal Use Case Name	Gerenciar Informações
Stereotype	Goal
AgentRoleActor	Gerenciador de Conteudo
Abstract	Este caso de uso interno descreve as etapas seguidas pelo agente Gerenciador de Conteúdo ao gerenciar as informações fornecidas pelo Desenvolvedor.
Initial Beliefs	Não é necessário validar informações relacionadas ao Objeto de Aprendizagem.
Perceptions	Sondagem das ações do desenvolvedor
Cenário Principal	
Ações do AgentRoleActor	
1. Executar o caso de uso interno "Perceber ações do desenvolvedor no sistema"	
Cenário Alternativo - Sondagem das ações do desenvolvedor percebida.	
Ações do AgentRoleActor	
1. Passar a acreditar que é necessário validar informações relacionadas ao Objeto de Aprendizagem.	
2. Tornar o objetivo uma intenção.	
3. Executar caso de uso Interno "Validar informações".	

Dessa forma, com o objetivo de preencher as lacunas identificadas tanto na revisão sistemática de literatura quanto na proposta existente da MASRML, planejamos desenvolver uma extensão da MASRML que abranja o diagrama de classes e suporte os conceitos necessários para o modelo BDI. Essa extensão será incorporada à visão de projeto do nosso processo de desenvolvimento de SMAs, que está em andamento. O processo atual se baseia no modelo da MASRML existente e abrange apenas a fase de engenharia de requisitos.

A tabela 8 apresenta uma comparação geral das características de cada linguagem detalhada no capítulo.

Tabela 8 – Comparação das Linguagens de Modelagem para Sistemas Multiagentes

Características das Linguagens	**MAS-ML**	**AUML**	**Notação proposta por Carla Silva**	**SEA-ML**	**MASRML**
Suporte ao modelo BDI	Sim, mas possui limitações quanto ao funcionamento	Não	Sim, mas possui limitações quanto ao funcionamento	Sim	Sim
Diagrama de Classes	Sim	Sim	Sim	Sim	Não
Representação de Percepções	Não é abordado no diagrama analisado	Sim	Não	Não	Sim
Representação de Planos	Sim, mas não possui classe própria	Não aborda	Sim	Sim, mas não possui classe própria	Sim
Representação de Objetivos	Sim, mas não possui uma classe própria	Não	Sim	Sim, mas não possui uma classe própria	Sim
Representação de Crenças	Sim, mas não possui uma classe própria	Sim	Sim	Sim, mas não possui uma classe própria	Sim
Representação de Comunicação entre agentes	Possui protocolo mas sem relacionamento entre classes de agente	Não, possui apenas um atributo onde é apresentado qual protocolo utilizar	Não	Não	Possui ações para representar comunicação

4.6 Proposta de Extensão da MASRML

Decidimos, neste trabalho, estender o metamodelo da UML, mais especificamente as metaclasses ”Classifier”, ”Operation” e ”Action”, com o objetivo de criar estereótipos que enriqueçam o metamodelo da MASRML. Essa decisão foi tomada levando em con-

sideração o fato de que a UML é uma linguagem de modelagem amplamente aceita e compreendida na área de engenharia de software, e que as extensões atuais da UML para SMAs apresentam lacunas identificadas em nosso estudo prévio.

A extensão proposta visa representar modelos conceituais similares aos produzidos pelo diagrama de classes padrão da UML, porém com o suporte a características específicas de SMAs. Essa extensão tem como diferencial o preenchimento das lacunas identificadas nas abordagens atuais, abordando os pontos destacados como importantes para a modelagem de SMAs.

Para melhorar a visualização e facilitar a explicação, optamos por organizar o metamodelo em diferentes visões, em que cada visão apresenta apenas as metaclasses relevantes para aquele contexto específico. Essa abordagem foi inspirada na linguagem SEA-ML, que também utiliza essa estratégia em sua notação.

A primeira visão, ilustrada na Figura 10, aborda o relacionamento entre agentes e papéis de agentes, além do envio de mensagens entre eles. Também são destacados os vínculos desses agentes e papéis de agentes com o ambiente e as organizações.

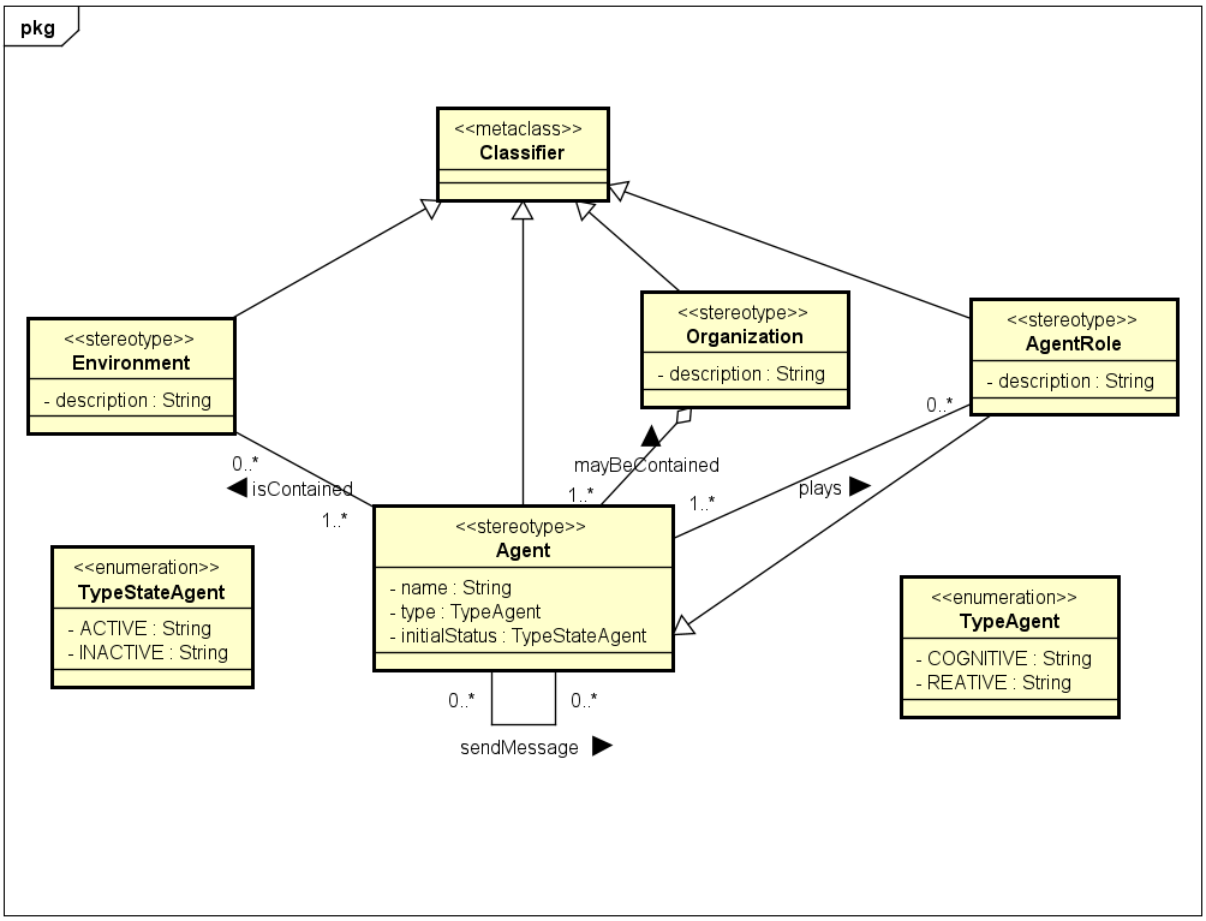


Figura 10 – Primeira Visão do metamodelo proposta para Extensão da MASRML.

Para representar o conceito de agente de *software*, foi criado o estereótipo «*Agent*». Esse estereótipo possui os atributos "*name*", "*type*" e "*initialStatus*". O atributo "*name*"

é do tipo *String* e foi adicionado para especificar o nome do agente. O atributo "*type*" é do tipo *TypeStateAgent*, uma enumeração que permite descrever se o agente é de tipo Cognitivo ou Reativo. Já o atributo "*initialStatus*" é do tipo *TypeStateAgent* e descreve o estado inicial do agente, podendo ser ativo ou dependente de uma ação do sistema para se tornar ativo.

Ao examinarmos o estereótipo «Agent», podemos observar que ele estabelece relacionamentos com outros estereótipos. Esses relacionamentos permitem representar que um agente está contido no ambiente, que um agente pode estar contido em uma organização e que um agente pode assumir um papel no sistema. Além disso, o estereótipo «Agent» possui uma associação reflexiva, permitindo representar o envio de mensagens entre agentes de software.

Embora alguns trabalhos utilizem apenas o conceito de papel de agente em seus modelos, como é o caso do metamodelo da SEA-ML (GETIR; CHALLENGER; KARDAS, 2014), o uso combinado dos conceitos de Agente e Papel de Agente pode trazer diversas vantagens para a modelagem de sistemas inteligentes. Ao permitir que o projetista represente ambos os conceitos, o modelo se torna mais flexível e capaz de representar com mais precisão a complexidade do sistema em questão.

Enquanto os agentes representam entidades que possuem características e comportamentos próprios, os papéis de agentes podem ser utilizados para representar diferentes funções que esses agentes podem assumir em um determinado contexto. Isso permite uma modelagem mais refinada e detalhada do comportamento do sistema.

Além disso, ao usar tanto Agent quanto Agent Role, a notação deve incluir uma descrição detalhada do papel de agente, incluindo quando o agente deve assumir a função, o propósito da função que ele assume e quaisquer outras informações relevantes. Essa descrição pode ajudar a esclarecer o papel de cada agente no sistema e garantir que ele esteja desempenhando suas funções de maneira adequada.

Dessa forma, ao combinar os conceitos de Agente e Papel de Agente em um modelo, é possível criar representações mais precisas e completas de sistemas inteligentes, tornando a modelagem mais eficiente e efetiva.

Com relação aos estereótipos «Environment» e «Organization», ambos são conceitos importantes para o uso de SMAs, todavia, como nosso foco inicial é detalhar um suporte ao modelo BDI e as características internas dos agentes, entendemos que uma visão minimalista dos conceitos, contendo apenas um atributo de descrição seria o mais adequado. Pretendemos, em trabalhos futuros, estender estas visões mostrando uma abrangência maior de características de relacionamento entre agentes e de agentes com o ambiente.

A segunda visão do metamodelo pode ser vista na Figura 11, esta visão representa as características gerais de um agente (ou papel de agente), nela podemos identificar as crenças, objetivos e percepções de um agente. Entretanto, está visão mostra apenas o

nome destas características, uma abordagem mais detalhada é descrita em outra visão.

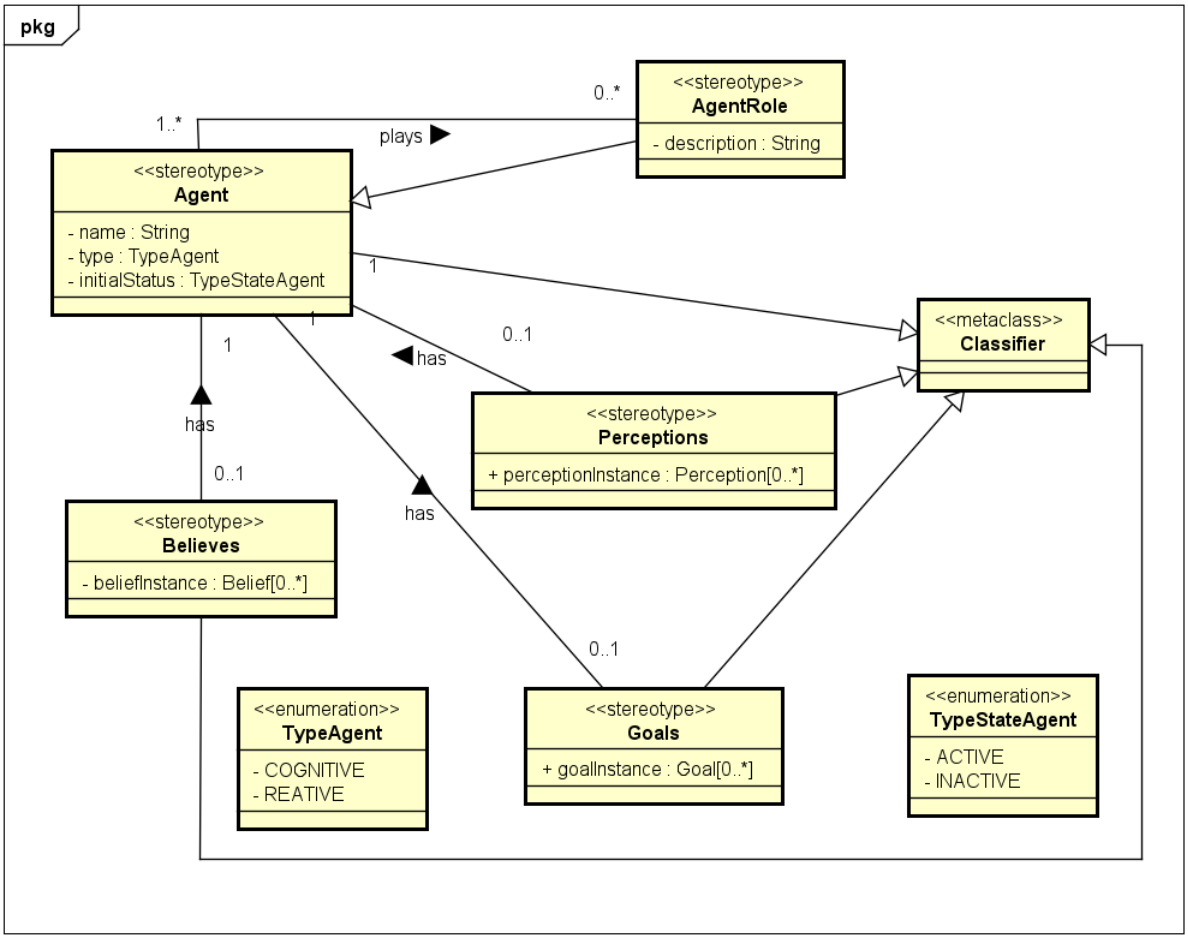


Figura 11 – Segunda Visão do metamodelo proposta para Extensão da MASRML.

Os estereótipos «*Perceptions*», «*Beliefs*» e «*Goals*» são especializações da metaclasse «*Classifier*» e foram criados para representar as características frequentemente presentes em agentes e/ou papéis de agentes. Eles seguem a mesma estratégia de funcionamento e, na prática, funcionam como uma lista de características desses agentes. Isso significa que um agente ou papel de agente pode ter zero ou várias percepções, crenças e objetivos, utilizando os atributos do tipo *Perception* [0..*], *Belief* [0..*] e *Goal* [0..*].

Essa abordagem simplifica a criação de modelos e evita a poluição visual, permitindo que o projetista crie listas claras e diretas de percepções, crenças e objetivos para cada agente ou papel de agente. Essa forma de representação facilita a compreensão do modelo, tornando-o mais acessível para outros membros da equipe envolvidos no projeto.

A terceira visão do Metamodelo, apresentada na Figura 12, detalha as características e as conexões de um objetivo, incluindo as percepções, crenças e o nome dos planos relacionados a ele.

O estereótipo «*Goal*» representa uma característica do Agente e possui os atributos "*status*" e "*description*". O atributo "status"do objetivo é do tipo "*StatusGoal*", uma

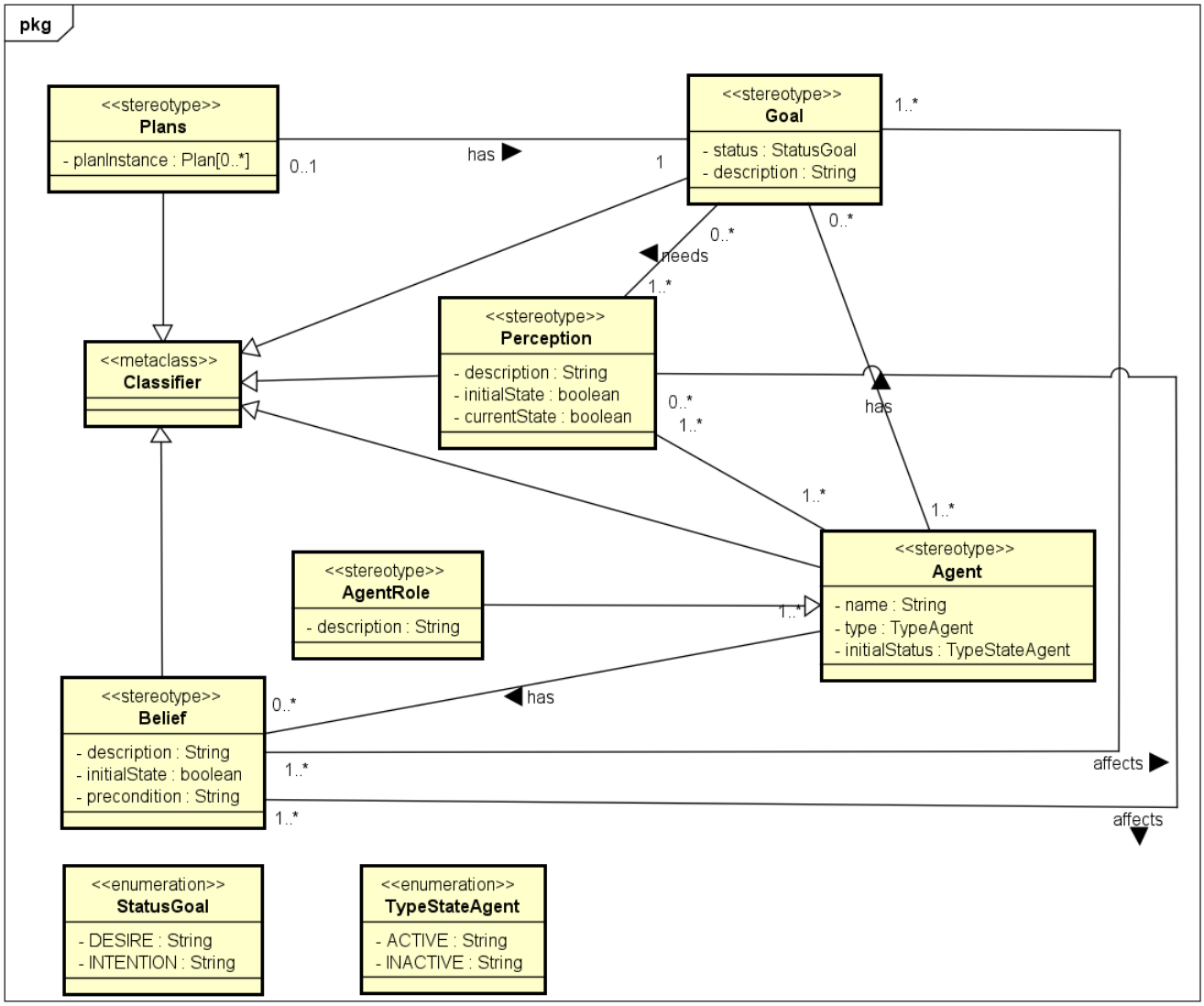


Figura 12 – Terceira Visão do metamodelo proposta para Extensão da MASRML.

enumeração que classifica o objetivo como Desejo (*Desire*) ou Intenção (*Intention*). O atributo "*description*" fornece um resumo do que o objetivo representa.

Esta visão também inclui o estereótipo «*Perception*». Ao examinarmos as conexões desse estereótipo, podemos notar que o objetivo depende da percepção para funcionar e que a percepção afeta as crenças do agente.

O estereótipo «Perception» possui três atributos: "*description*", "*initialState*" e "*currentState*". O atributo "*description*" tem como objetivo resumir o funcionamento da percepção, mostrando o que ela monitora e quais resultados podem acioná-la. O atributo "*initialState*" indica se a percepção deve ser ativada ou desativada no início do sistema, enquanto o atributo "*currentState*" informa se a percepção está ativada ou desativada no estado atual do sistema.

O estereótipo «*Belief*» é uma característica do agente que é afetada pelas percepções e pode afetar os objetivos. O «*Belief*» possui os atributos "*description*", "*initialS-*

tate" e "*precondition*". O atributo "*description*" permite descrever a crença da forma considerada mais adequada pelo projetista. O atributo "*initialState*" determina se o agente inicialmente acredita ou não na crença em questão. Por último, o atributo "*precondition*" classifica em que momento o agente passa a acreditar que essa crença é verdadeira ou falsa.

Para finalizar esta visão, o estereótipo «*Plans*» segue a mesma abordagem de lista da Visão 2 (Figura 11), em que a metaclasse nos permite representar 0 ou muitas instâncias de um Plano, mostrando apenas o nome desse plano.

A última visão presente neste metamodelo pode ser vista na Figura 13. Essa visão tem como objetivo detalhar o funcionamento de um plano.

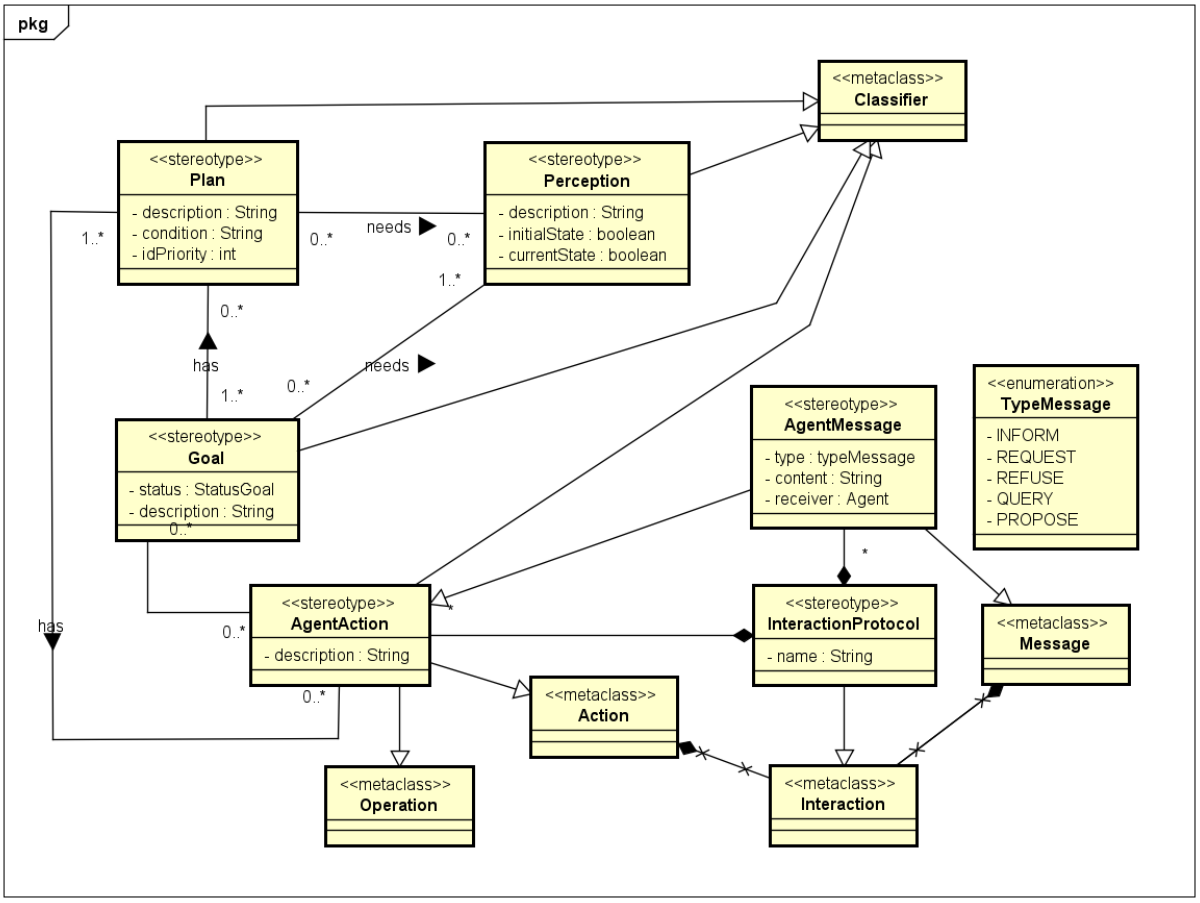


Figura 13 – Quarta Visão do metamodelo proposta para Extensão da MASRML.

O estereótipo «Plan» deve estar contido em um objetivo e pode possuir ações de agente («AgentAction»). O «Plan» possui os atributos description, condition e idPriority. O atributo description deve conter um resumo do funcionamento do plano. O atributo condition deve detalhar em que momento o plano deve ser disparado. E por último, o atributo idPriority deve apresentar a posição do plano em uma ordem de prioridade de execução, caso haja vários planos a serem executados.

O estereótipo «AgentAction» é uma especialização de «Operation» e tem como objetivo detalhar uma ação específica do agente ou papel no ambiente. Essa ação pode ser uma mensagem descrita pela classe «AgentMessage», que também é uma especialização de «AgentAction». A «AgentMessage» possui os atributos typeMessage, content e receiver. O atributo typeMessage classifica o tipo de mensagem que um agente pode enviar, utilizando a classe enumeração «TypeMessage» que define os tipos inform, request, refuse, query e propose conforme o padrão FIPA. O atributo content contém o conteúdo da mensagem enviada, enquanto o atributo receiver informa o agente que deve receber a mensagem.

Um plano pode conter interações mais complexas e, caso seja necessário descrever essas interações, utiliza-se o estereótipo «InteractionProtocol», que é uma extensão da metaclasses «Interaction». O atributo «name» desse estereótipo é utilizado para descrever o protocolo de interação necessário para a interação dos agentes.

O diagrama geral do metamodelo, apresentando as quatro visões, pode ser visualizado na Figura 14.

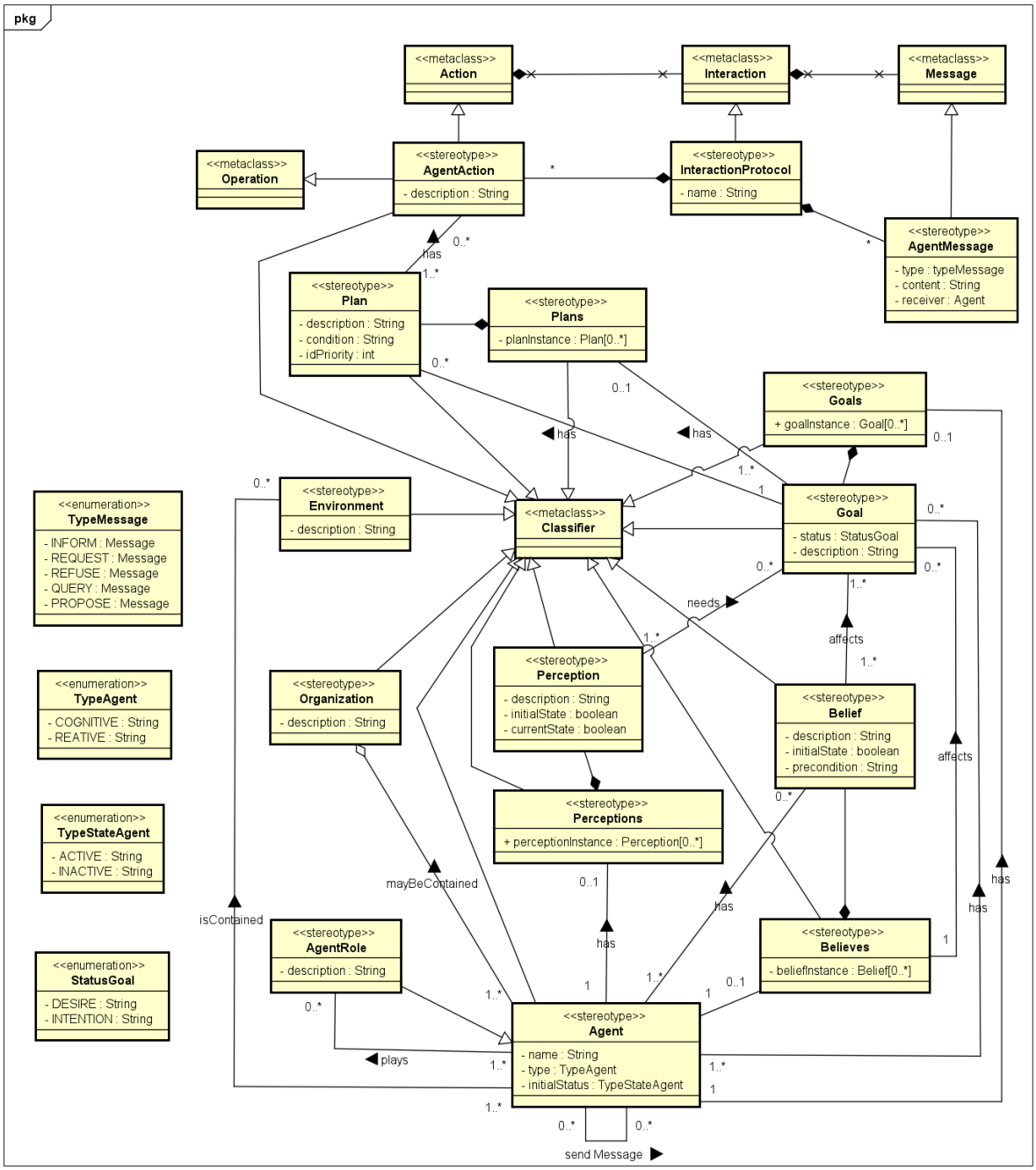


Figura 14 – Visão Geral do metamodelo proposta para Extensão da MASRML.

5 AVALIAÇÃO DA EXTENSÃO

Este capítulo tem como objetivo apresentar detalhadamente o experimento realizado, juntamente com os resultados obtidos após a sua finalização. A estrutura do capítulo é a seguinte: na Seção 5.1, é apresentada uma descrição do experimento realizado, incluindo detalhes sobre a metodologia utilizada. Na Seção 5.2, são apresentados os objetivos do experimento, delineando claramente o que se pretende alcançar. Na Seção 5.3, são apresentadas as questões de pesquisa que nortearam o experimento, juntamente com uma descrição de sua importância para a investigação. Na Seção 5.4, são apresentados os dados coletados para a avaliação quantitativa. Por fim, na Seção 5.5, são apresentados os resultados obtidos com o experimento, incluindo a resposta para as questões de pesquisa formuladas.

5.1 Introdução ao Experimento

No experimento realizado, buscamos explorar os pontos fortes e fracos da proposta de extensão da MASRML por meio da comparação de modelos. O foco principal foi o desenvolvimento de modelos do diagrama de classes pelos participantes, utilizando como base o metamodelo proposto.

Destacamos também que, embora não tenhamos seguido um protocolo de experimento exato, nos inspiramos no protocolo experimental descrito por Wohlin et al. (2012) como referência e base para a realização do nosso experimento.

Durante o experimento, foram coletados os modelos criados pelos participantes, os quais foram comparados com o resultado esperado. Essa comparação foi feita tanto de forma quantitativa, verificando o número de classes e atributos identificados nos modelos, quanto de forma qualitativa, analisando se os modelos desenvolvidos representavam adequadamente o que era esperado.

Essa abordagem permitiu identificar os pontos fortes da proposta de extensão da MASRML, destacando os aspectos em que os participantes foram capazes de criar modelos precisos e completos, conforme o metamodelo proposto. Além disso, também foi possível identificar os pontos fracos, indicando as áreas em que houve dificuldade na compreensão ou aplicação do metamodelo.

Essa análise comparativa entre os modelos desenvolvidos e o resultado esperado proporcionou *insights* valiosos sobre a eficácia e usabilidade da extensão proposta da MASRML, contribuindo para refinamentos futuros e aprimoramento da abordagem.

O experimento contou com a participação de 9 indivíduos. Entre os participantes, a maioria estava cursando ou já havia concluído o mestrado. Os dados de formação acadêmica indicaram que 33,3% dos participantes estavam atualmente cursando o mestrado, enquanto 22,2% já haviam obtido o título de mestre. Além disso, 11,1% eram doutorandos, 11,1% estavam na graduação e 22,2% já haviam concluído a graduação.

Todos os participantes possuíam algum nível de conhecimento acadêmico em Agen-

tes de Software, totalizando 100% do grupo. Em relação à modelagem UML, 66,7% dos participantes tinham conhecimento acadêmico sobre o assunto, enquanto 33,3% possuíam experiência profissional relacionada à modelagem UML.

Essas informações sobre o perfil dos participantes são relevantes para compreender a base de conhecimento e experiência que eles possuíam ao realizar o experimento. Todas as informações coletadas sobre o perfil dos participantes podem ser vistas no anexo B

Com base no perfil dos participantes, é possível inferir que o grupo é composto predominantemente por indivíduos com formação acadêmica, abrangendo diferentes níveis educacionais, como graduação, mestrado e doutorado. Além disso, os participantes possuem diversos níveis de experiência em modelagem UML, incluindo tanto experiência acadêmica quanto profissional.

Essa diversidade no perfil dos participantes pode ser interessante para avaliar a efetividade do modelo proposto em diferentes níveis de conhecimento sobre agentes de software e diferentes níveis de experiência em modelagem UML. No entanto, é importante notar que a amostra é pequena, o que pode limitar a generalização dos resultados obtidos.

Antes de iniciar o experimento, apresentamos aos participantes uma aula introdutória ao tema. Essa aula teve a duração de 20 minutos e abordou os seguintes temas: SMAs, características internas dos agentes, Modelo BDI, Diagramas de Classes, Metamodelos da UML, Extensão da MASRML e Visões do Metamodelo proposto.

Após a aula introdutória, foi realizada uma sessão de perguntas para esclarecer eventuais dúvidas dos participantes. Foi enfatizado que durante o experimento, poderíamos responder perguntas relacionadas ao cenário proposto, mas não auxiliaríamos no desenvolvimento dos modelos.

Com o encerramento das perguntas, enviamos aos participantes um documento contendo as instruções para o experimento e a descrição dos cenários do sistema proposto. Este documento pode ser visualizado no Anexo das Instruções (A).

É importante destacar que não delimitamos um tempo exato de término do experimento, mas os tempos variaram de 58 minutos a 1 hora e 44 minutos. Deste modo, este capítulo tem como objetivo apresentar o planejamento do experimento, bem como os resultados obtidos.

5.2 Objetivo do Experimento

O objetivo deste experimento é obter evidências sobre a eficácia do uso da extensão da MASRML, na qual busca-se adaptar e estender o Diagrama de Classes no contexto de SMAs. Para isso, foi necessário realizar a comparação entre os modelos criados pelos participantes do experimento e o modelo esperado pelos avaliadores.

A extensão da MASRML, por meio da adição do Diagrama de Classes adaptado para o contexto de SMAs, permite a especificação dos aspectos de agência, como papéis, objetivos, crenças e intenções dos agentes, além das relações entre eles.

Por meio deste experimento, procuramos verificar se o uso da extensão da MASRML pode ser eficaz para a modelagem de SMAs. Caso os resultados indiquem uma maior eficácia, isso poderá contribuir para o desenvolvimento de novas abordagens e metodologias de desenvolvimento de SMAs mais precisas e eficientes.

5.3 Questões de Pesquisa

As questões de pesquisa têm um status importante como o eixo do processo de pesquisa. Essas questões ajudam a vincular a revisão da literatura do pesquisador aos tipos de dados que serão coletados (BRYMAN, 2007). Nossas questões de pesquisa também têm como objetivo apresentar uma avaliação qualitativa dos modelos produzidos pelos participantes do experimento.

Para garantir a validade e confiabilidade de um experimento de avaliação, é essencial que as questões de pesquisa estejam claramente definidas. No caso do nosso experimento de comparação de modelos, o objetivo foi avaliar o desempenho dos participantes na criação de modelos e compará-los a um modelo de referência. Isso requeriu que as questões de pesquisa fossem formuladas de forma clara e objetiva, para identificar as diferenças, semelhanças e pontos fortes e fracos dos modelos produzidos pelos participantes em comparação com o modelo de referência produzido pelo avaliador. As questões de pesquisa formuladas foram as seguintes:

- **QP1:** Qual é a precisão dos modelos criados em comparação com a solução desenvolvida pelos avaliadores?
  - **Hipótese alternativa 1:** Os modelos criados são parcialmente precisos em comparação com a solução desenvolvida pelos avaliadores. Alguns aspectos dos modelos criados podem ser precisos, enquanto outros podem não estar em conformidade com a solução desenvolvida pelos avaliadores;
  - **Hipótese alternativa 2:** Os modelos criados não são precisos em comparação com a solução desenvolvida pelos avaliadores.
  - **Hipótese nula:** Não há diferença significativa entre os modelos produzidos pelos participantes e o metamodelo base em termos de compreensão e eficácia na modelagem de requisitos.
- **QP2:** Quão bem os modelos descrevem os dados de características dos agentes BDI?
  - **Hipótese alternativa 1:** Os modelos criados pelos participantes descrevem adequadamente as características do modelo BDI;
  - **Hipótese alternativa 2:** Os modelos criados não são capazes de descrever adequadamente as características do modelo BDI.

  - **Hipótese nula:** Não há diferença significativa entre os modelos produzidos pelos participantes em relação à descrição dos dados de entrada e à percepção das características dos agentes BDI.

- **QP3:** Os modelos criados seguem corretamente as definições e restrições do metamodelo?

  - **Hipótese alternativa 1:** Os modelos criados seguem corretamente as definições e restrições do metamodelo;
  - **Hipótese alternativa 2:** As definições e/ou restrições do metamodelo são ignoradas ou não são atendidas adequadamente nos modelos criados.
  - **Hipótese nula:** Não há diferença significativa entre os modelos produzidos pelos participantes em relação à conformidade com as definições e restrições do metamodelo.

- **QP4:** O projeto criado segue o padrão esperado de 4 modelos com visões específicas de funcionamento?

  - **Hipótese alternativa 1:** Alguns participantes seguem o projeto com as 4 visões corretas com funções específicas e outros participantes não seguem;
  - **Hipótese alternativa 2:** Os projetos não seguem o padrão esperado de 4 modelos com visões específicas de funcionamento.
  - **Hipótese nula:** Não há diferença significativa entre os projetos dos participantes em relação ao padrão esperado de 4 modelos com visões específicas de funcionamento.

5.4 Dados para a Avaliação Quantitativa

Para obter uma base sólida de dados para responder às questões qualitativas, detalhamos uma série de questões quantitativas para extração nos modelos criados pelos participantes. A definição dessas questões tem como objetivo auxiliar na resposta das questões de pesquisa.

A Tabela 9 apresenta a questão e o número esperado como resposta para a avaliação.

Questão para Extração	**Resultado Esperado**
Número de Classes criadas na Visão 1	4 Classes (Agente Pedestre, Agente Veículo Autônomo, Agente de Trânsito e Classe de ambiente)
Número de Classes criadas na Visão 2	12 Classes (3 Classes de Agentes, 3 Classes de Crenças, 3 Classes de Objetivos e 3 Classes de Percepções)
Número de Classes criadas na Visão 3	5 classes (1 Classe para Agente/Papel de agente, 1 Classe para objetivo, 1 Classe para percepção, 1 Classe para crença e 1 Classe para planos)
Número de Classes criadas na Visão 4	5 Classes (1 Classe para objetivo, 1 Classe para Plano, 2 Classes para ações e 1 Classe para mensagem da Ação)
Quantidade de Relacionamento entre agentes	2
Quantidade de Crenças dos Agentes	Agente Veículo Autônomo = 3 crenças Agente Pedestre = 1 crença Agente de Trânsito = 2 crenças
Quantidade de Percepções dos Agentes	Agente Veículo Autônomo = 2 percepções Agente Pedestre = 1 percepção Agente de Trânsito = 3 percepções
Quantidade de Objetivos dos Agentes	Agente Veículo Autônomo = 2 objetivos Agente Pedestre = 1 objetivo Agente de Trânsito = 2 objetivos
Quantidade de Planos do Agente de Trânsito	1
Quantidade de Ações relacionadas ao Plano Garantir Segurança dos Pedestre e Veículos	2
Quantidade de Mensagens Relacionadas ao Plano Garantir Segurança dos Pedestre e Veículos	1

Tabela 9 – Dados extraídos dos modelo produzidos pelos Participantes.

5.5 Resultados Obtidos

Esta Seção tem como objetivo apresentar de forma gráfica e textual os resultados obtidos com a avaliação quantitativa dos modelos.

Comparando os dados apresentados na Figura 15, é possível observar que a maioria dos participantes conseguiu atingir o resultado esperado de 4 classes na Visão 1 e de 12 classes na Visão 2. Já na Visão 3 e 4, os resultados foram mais variados e alguns participantes não obtiveram um resultado totalmente correto.

Acreditamos que isto seria esperado, visto que as visões 3 e 4 focam em temas mais complexos como as características que envolvem o objetivo e o plano do agente. Entretanto, mesmo nessas visões, 66,67% dos participantes identificaram acima da metade do número esperado de classes na visão 3 e 88,89 na visão 4, sendo que somente 3 participantes não conseguiram um mínimo de 3 classes na visão 3 e apenas 1 na visão 4.

Em geral, podemos observar que a maioria dos participantes conseguiu atingir o resultado esperado em algumas das visões do metamodelo, mas nenhum deles conseguiu atingir o resultado esperado em todas as visões. Isso pode indicar que, apesar da proposta cumprir o seu objetivo de modelar as características próprias de SMAs por meio do diagrama de classes, ainda existem aspectos que podem ser aprimorados. Também é

importante considerar que a falta de experiência dos participantes com a linguagem proposta pode ter influenciado nos resultados. Acreditamos que uma segunda aplicação da linguagem, em um problema de complexidade semelhante e com os mesmos participantes, poderia resultar em melhores resultados.

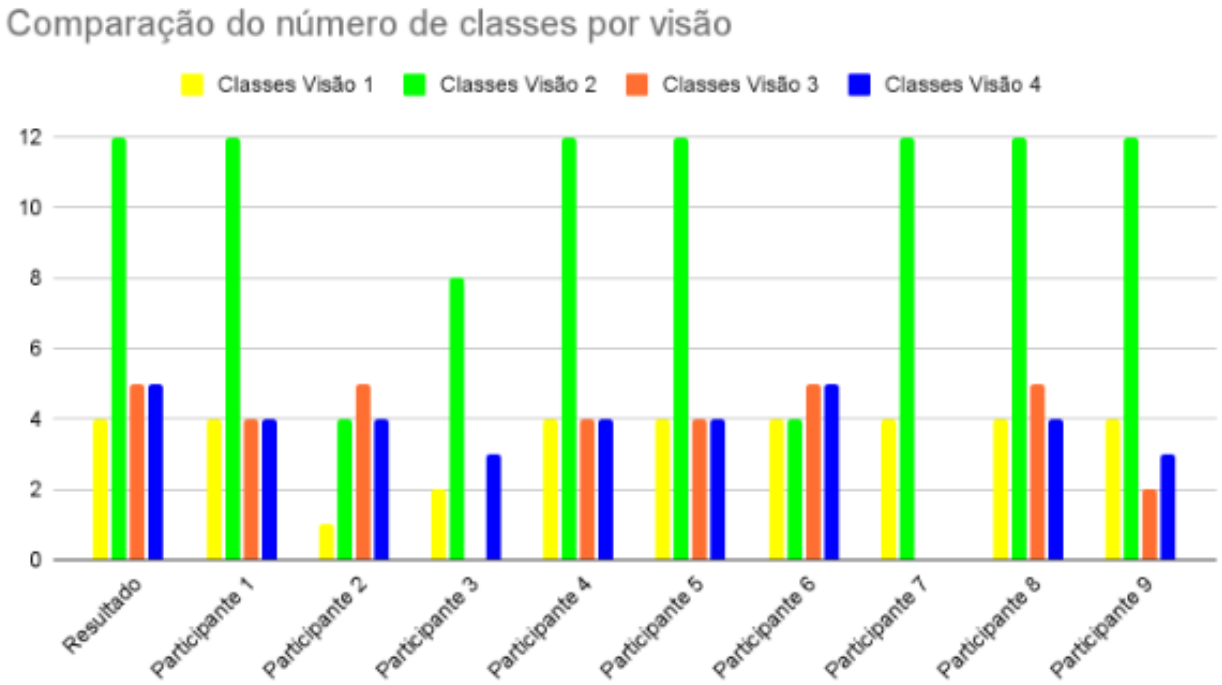


Figura 15 – Gráfico de Comparação do número de Classes criadas pelos participantes por Visão do Metamodelo.

A Figura 16 apresenta os resultados obtidos com relação ao número de relacionamentos entre agentes identificados. Avaliou-se o número de relacionamentos entre agentes gerados por cada modelo produzido pelos participantes. O resultado esperado era de 2 relacionamentos e os resultados obtidos mostraram que 55,6% dos participantes identificaram esses 2 relacionamentos. Ou seja, a maioria dos participantes obteve um resultado totalmente satisfatório.

Por outro lado, 33,3% dos participantes produziram apenas 1 relacionamento, enquanto 11,1% não identificaram nenhum relacionamento. Esses resultados indicam que, embora a maioria tenha conseguido atingir os resultados esperados, um número razoável encontrou apenas metade e um pequeno número não identificou relacionamentos entre os agentes.

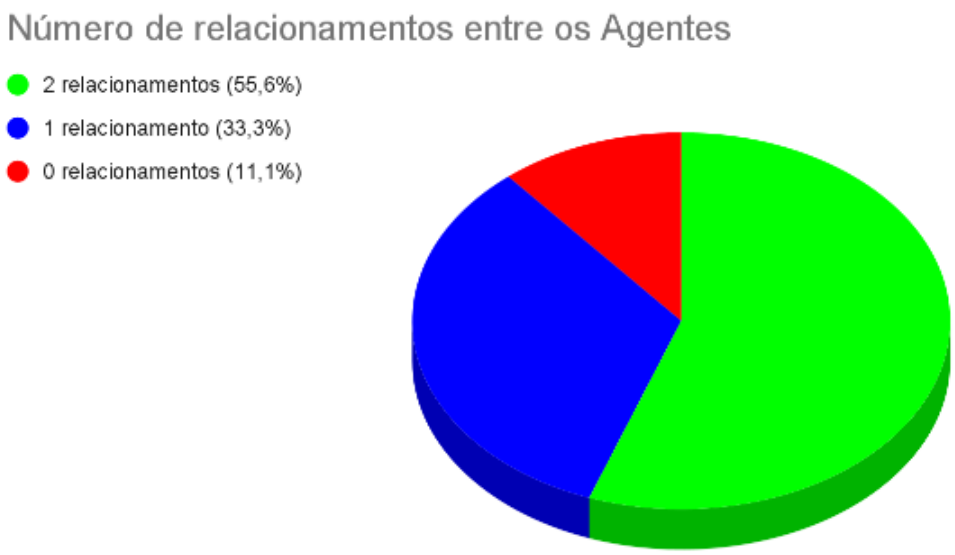


Figura 16 – Gráfico de Número de relacionamentos, entre agentes, criados pelos participantes.

A Figura 17 apresenta o resultado em relação à quantidade de planos identificados para o Agente de Trânsito. O gráfico divide os resultados em três categorias: (I) número correto identificado, (II) número correto, porém com erros de diagramação, e (III) número incorreto.

Figura 17 – Gráfico de número de Planos do agente de Trânsito criados pelos participantes.

Os resultados indicam que mais da metade (55,5%) dos participantes conseguiram identificar corretamente o número de planos propostos. Como um todo, 33,3% dos parti-

cipantes alcançaram o número correto, sem problemas na forma de identificação do plano, e 22,2% identificaram corretamente, porém apresentaram problemas de diagramação. Por fim, 44,4% não alcançaram o número correto de planos, o que pode indicar uma maior dificuldade na compreensão do cenário proposto ou na aplicação dos conceitos de planos de agentes.

Outro aspecto avaliado foi o número de ações geradas por cada modelo produzido pelos participantes. O resultado pode ser visto na Figura 18. O esperado era que cada modelo apresentasse 2 ações, no entanto, os resultados obtidos pelos participantes foram diferentes do esperado.

Nenhum dos participantes conseguiu gerar o número esperado de 2 ações, com 77,8% deles apresentando apenas 1 ação em seus modelos. Já 22,2% dos participantes não geraram ações em seus modelos.

Esses resultados sugerem que os participantes tiveram dificuldades em identificar e implementar as ações necessárias para cada agente em seus modelos. É possível que tenha havido falta de clareza no cenário descrito, pois os participantes entenderam a forma de diagramação das ações (já que a grande maioria gerou 1 ação), mas nenhum deles gerou duas ações.

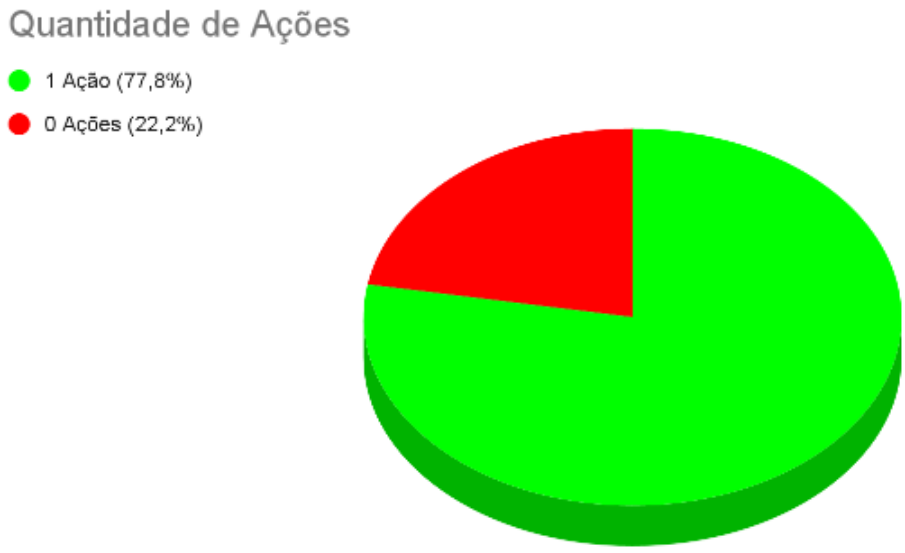


Figura 18 – Gráfico de Quantidade Ações dos agentes criados pelos participantes.

Outro ponto avaliado foi o número de mensagens geradas por cada modelo produzido pelos participantes, um aspecto crucial em muitos SMAs, pois é por meio das mensagens que os agentes podem comunicar informações relevantes entre si. O resultado obtido pode ser visto na Figura 19, em que o esperado era que cada modelo gerasse uma única mensagem. Dos participantes, 80% gerou exatamente uma mensagem conforme esperado, enquanto 20% não gerou mensagens.

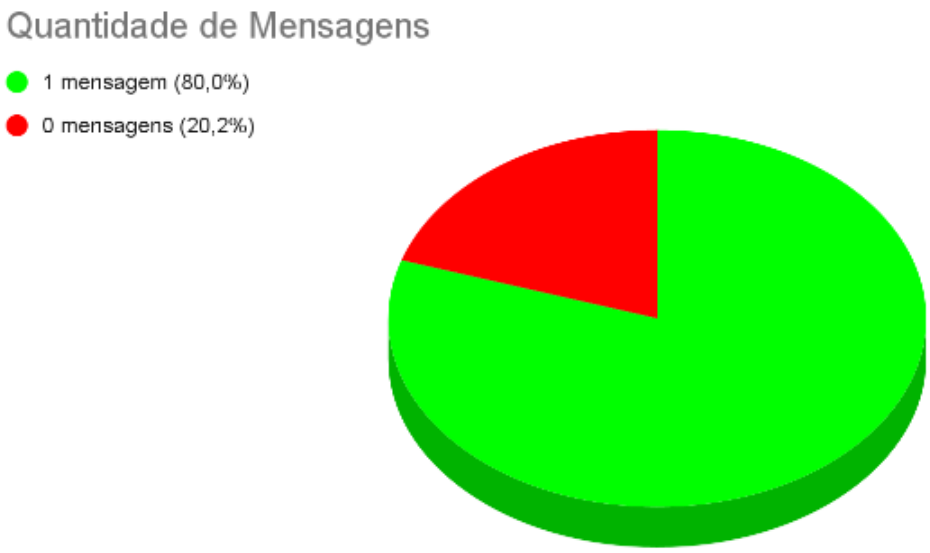


Figura 19 – Gráfico de Quantidade de Mensagens dos Agentes criados pelos participantes.

As características de crenças, percepções e objetivos receberam destaque em nossa avaliação, pois são essenciais no desenvolvimento de agentes BDI. As percepções podem modificar as crenças dos agentes, que, por sua vez, podem afetar os objetivos, transformando-os em intenções. Dessa forma, analisamos como os participantes identificaram essas características dos agentes, que foram modeladas na segunda visão de nossa proposta.

Os resultados com relação à quantidade de crenças dos agentes podem ser vistos na Figura 20. Observa-se que os números referentes às crenças dos agentes são baixos, o que significa que os participantes têm maior facilidade em acertar ou errar todas as crenças, como é o caso do participante 1, que conseguiu identificar todas as crenças para os 3 agentes, e dos participantes 2 e 6, que identificaram apenas as crenças do Agente de Trânsito.

A partir disso, apresentamos a análise percentual das crenças identificadas por agente:

Crenças do Agente Veículo: (Esperado: identificação de 3 crenças).

- 22,2% dos participantes não identificaram nenhuma crença relacionada ao veículo.
- 22,2% dos participantes identificaram apenas 1 crença relacionada ao veículo.
- 33,3% dos participantes identificaram 2 crenças relacionadas ao veículo.
- 33,3% dos participantes identificaram as 3 crenças relacionadas ao veículo.

Crenças do Agente Pedestre: (Esperado: identificação de 1 crença).

- 33,3% dos participantes não identificaram nenhuma crença relacionada ao pedestre.

- 66,67% dos participantes identificaram apenas 1 crença relacionada ao pedestre.

Crenças do Agente de Trânsito (Esperado: identificação de 2 crenças).

- 0% dos participantes não identificaram nenhuma crença relacionada ao agente de trânsito.
- 22,2% dos participantes identificaram apenas 1 crença relacionada ao agente de trânsito.
- 88,89% dos participantes identificaram 2 crenças relacionadas ao agente de trânsito.

A partir disso, podemos observar que para todos os agentes, os participantes conseguiram identificar mais da metade das crenças. Portanto, entendemos que os resultados indicam que os participantes tiveram um bom desempenho na modelagem das crenças dos agentes de software com base no modelo BDI, especialmente no caso das crenças do agente de trânsito, que teve o melhor número de crenças identificadas. O fato de 88,89% identificarem o número esperado pode indicar que os participantes conseguiram entender e aplicar adequadamente o uso de crenças dos agentes na modelagem da MASRML.

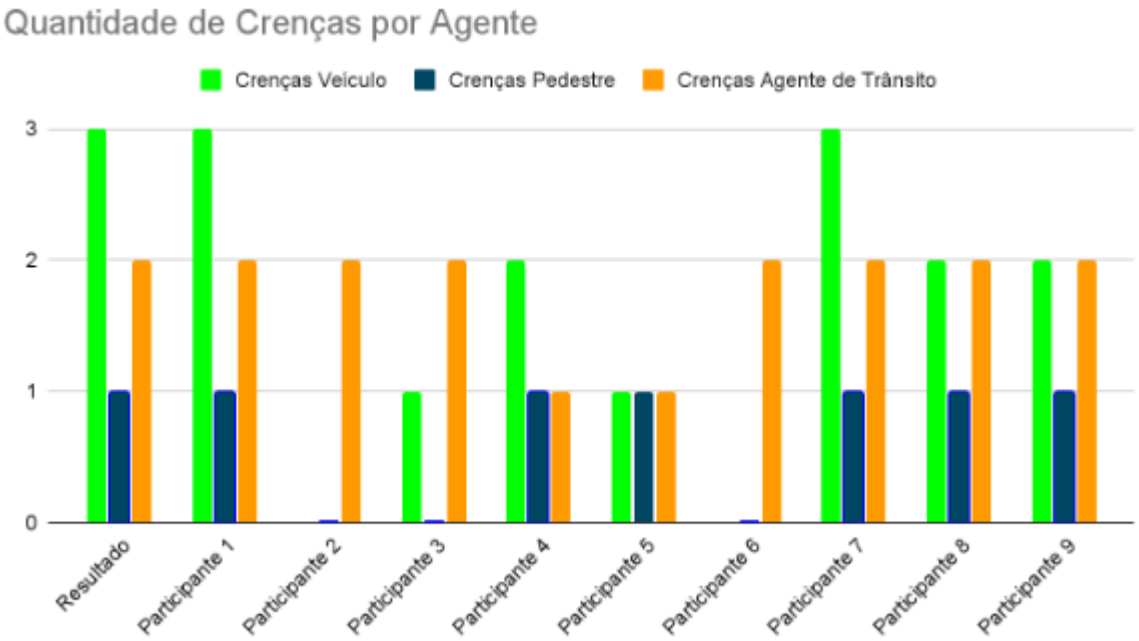


Figura 20 – Gráfico de Quantidade de Crenças por Agente criadas pelos Participantes.

Os resultados das quantidades de percepções identificadas pode ser visto na Figura 21. Para as percepções relacionadas aos agentes no cenário apresentado, temos a seguinte análise de porcentagem:

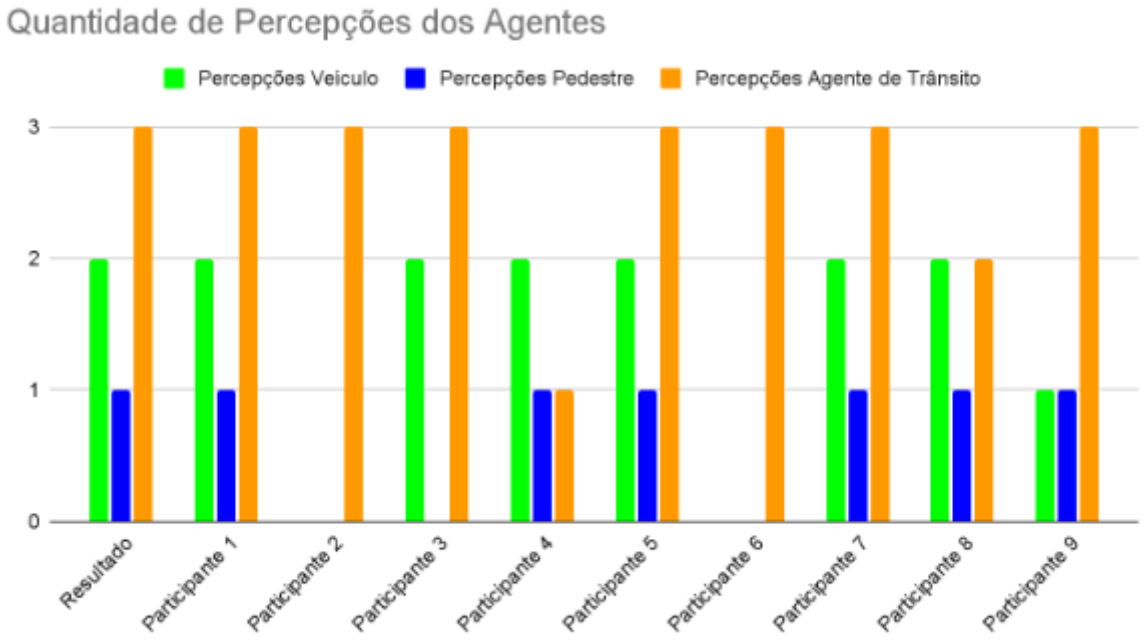


Figura 21 – Gráfico de Quantidade de Percepções por Agente criadas pelos Participantes.

Agente Veículo: (Esperado: identificação de 2 percepções).

- 0 percepções: 22,2% dos participantes
- 1 percepção: 11,1% dos participantes
- 2 percepções: 66,67% dos participantes

Agente Pedestre: (Esperado: identificação de 1 percepção).

- 0 percepções: 33,33% dos participantes
- 1 percepção: 66,67% dos participantes

Agente de Trânsito: (Esperado: identificação de 3 percepções).

- 0 percepções: 0% dos participantes
- 1 percepção: 11,1% dos participantes
- 2 percepções: 11,1% dos participantes
- 3 percepções: 77,78% dos participantes

A maioria dos participantes conseguiu identificar pelo menos uma percepção para cada um dos agentes, mesmo que não tenha sido a quantidade esperada. Para o Agente de Trânsito, que era o agente com a maior quantidade esperada de percepções, a maioria dos

participantes conseguiu identificar 3 percepções, que era o número esperado. O mesmo acontece para os Agentes Pedestre e Veículo, para os quais a maioria dos participantes conseguiu identificar a quantidade esperada de percepções.

Por último, os resultados com relação aos objetivos identificados podem ser vistos na Figura 22. Desta forma, analisando os dados apresentados, temos as seguintes porcentagens de identificação de objetivos:

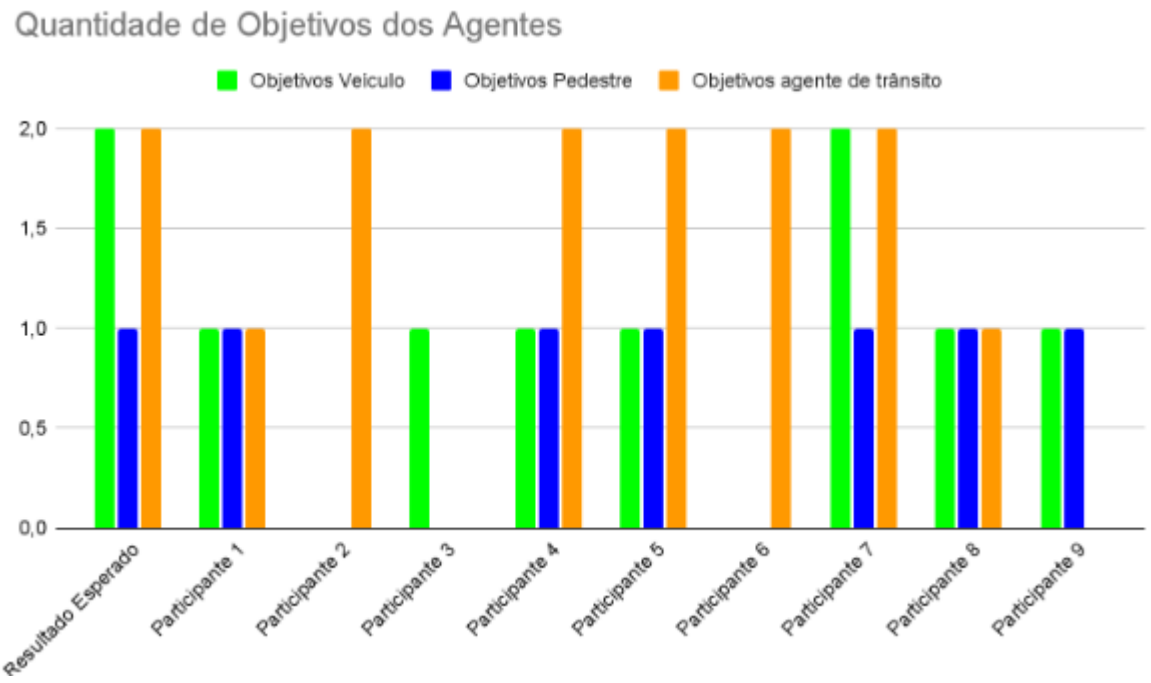


Figura 22 – Gráfico de Quantidade de Objetivos por Agente criados pelos Participantes.

Agente Veículo: (Esperado: identificação de 2 objetivos).

- 0 objetivos: 22,22% dos participantes
- 1 objetivo: 66,67% dos participantes
- 2 objetivos: 11,11% dos participantes

Agente Pedestre: (Esperado: identificação de 1 objetivo).

- 0 objetivos: 33,33% dos participantes
- 1 objetivo: 66,67% dos participantes

Agente de Trânsito: (Esperado: identificação de 2 objetivos).

- 0 objetivos: 22,22% dos participantes
- 1 objetivo: 22,22% dos participantes
- 2 objetivos: 55,56% dos participantes

Os dados sobre os objetivos dos agentes apresentaram valores mais baixos em comparação às crenças e percepções. No entanto, apenas o Agente Veículo teve um número de objetivos identificados menor que a metade do esperado. Acreditamos que isso se deve ao fato de um dos objetivos do agente veículo estar mencionado de forma dispersa no texto, o que levou apenas o participante 7 a identificá-lo. No entanto, sete dos nove participantes conseguiram identificar o outro objetivo, o que nos leva a crer que o problema principal reside na descrição do cenário e não na dificuldade de diagramação.

Com relação aos objetivos do Agente Pedestre e do Agente de Trânsito, acreditamos que eles foram identificados satisfatoriamente pelos participantes, o que demonstra que eles conseguiram compreender corretamente os objetivos desses agentes.

5.6 Respostas para as Questões de Pesquisa

Esta seção tem como objetivo apresentar as respostas para as Questões de Pesquisa. Para as questões 1 e 2, as hipóteses foram testadas por meio de uma análise da precisão dos resultados dos modelos em relação à solução desenvolvida pelos avaliadores e pelos resultados obtidos na avaliação quantitativa dos modelos. Para as questões 3 e 4, as hipóteses foram testadas a partir de uma análise de conformidade do modelo em relação às definições e restrições do metamodelo, bem como das explicações de seu funcionamento.

Q1: Quão preciso os modelos criados foram considerados em comparação com a solução desenvolvida pelos avaliadores?

Resposta: Com base na avaliação quantitativa dos dados e nas estatísticas geradas, é possível afirmar que a hipótese formulada neste experimento foi parcialmente confirmada. Os modelos criados pelos participantes apresentaram resultados diversos em relação à solução desenvolvida pelos avaliadores, indicando que a precisão dos modelos é limitada em alguns aspectos.

Embora alguns aspectos dos modelos tenham apresentado acurácia, como a quantidade de mensagens geradas pelos agentes, outros aspectos, como o número de ações do agente, não estavam totalmente em conformidade com a solução desenvolvida pelos avaliadores. Esses resultados indicam que os modelos criados pelos participantes podem ser úteis em alguns contextos, mas em outros precisam ser aprimorados.

No entanto, é importante destacar que este foi um experimento piloto, com um número limitado de participantes e um escopo limitado de análise. Portanto, são necessários mais experimentos para um entendimento mais aprofundado da acurácia dos modelos criados pelos participantes. É possível que fatores como o tempo de experiência dos participantes na modelagem de SMAs, a complexidade do problema proposto e a qualidade da documentação fornecida possam afetar a acurácia dos modelos.

Q2: Quão bem os modelos descrevem os dados de características dos agentes BDI?

Resposta: Com base nos resultados apresentados, é possível afirmar que os mo-

delos criados pelos participantes descrevem adequadamente as características do modelo BDI; Em nossa avaliação quantitativa, avaliamos o número de crenças, percepções e objetivos identificados pelos participantes. Essas características são essenciais para a modelagem de agentes BDI. Os dados mostraram que é possível perceber claramente as características dos agentes BDI nos modelos criados. O número de acertos apresentado pelos participantes nos mostra que os modelos foram capazes de capturar as características esperadas dos agentes BDI.

No entanto, é importante destacar que alguns modelos apresentaram problemas, indicando que em alguns casos os participantes podem ter falhado em identificar ou modelar corretamente algumas características dos agentes BDI. Ainda assim, esses erros não foram significativos e podem ter sido influenciados pelo tempo e complexidade da descrição dos cenários.

Em resumo, os resultados indicam que os modelos criados pelos participantes foram capazes de descrever com precisão os dados de entrada, e que as características dos agentes BDI são claramente perceptíveis nos modelos criados. Isso sugere que os participantes tiveram um bom entendimento dos conceitos de crenças, percepções e objetivos dos agentes BDI, e foram capazes de aplicá-los com sucesso na criação de modelos.

QP3: Os modelos criados seguem corretamente as definições e restrições do metamodelo?

Resposta: Para esta questão, separamos os resultados obtidos por visão do metamodelo. Porém, em relação à hipótese formulada, acreditamos que os modelos criados seguiram parcialmente as definições e restrições do metamodelo.

Visão 1: Os resultados do experimento indicaram que 6 dos participantes conseguiram desenvolver um modelo alinhado ao padrão esperado, o que reflete um bom nível de compreensão e habilidade na aplicação do padrão.

Contudo, foi observado que 3 participantes apresentaram algumas discrepâncias em seus modelos. Dois deles esqueceram de incluir o atributo "description", que é relevante para a compreensão do propósito de cada classe. Além disso, esses mesmos 3 participantes não utilizaram o atributo "initialStatus", que pode auxiliar na definição do estado inicial dos agentes no sistema.

Outros equívocos incluíram a falta do uso do atributo ”typeAgent” por um dos participantes e a não criação da classe de ambiente por outro. Além desses erros, houveram participantes que utilizaram setas de relacionamento de forma inadequada, mas que foram considerados erros menores. Não iremos focar em detalhar estes erros menores pois consideramos que não afetaram significativamente a avaliação, visto que são erros de modelagem simples e não refletem os aspectos referentes a utilização do metamodelo.

Visão 2: Constatou-se que 7 dos participantes atenderam completamente ao padrão esperado. Entretanto, dois participantes apresentaram erros em seus modelos. Um deles acrescentou classes que não deveriam estar presentes no diagrama, além de não

ter direcionado os relacionamentos entre as classes e esquecido de utilizar os estereótipos nas classes. Já o outro participante cometeu apenas o erro de não ter direcionado os relacionamentos entre as classes.

Apesar dos erros cometidos por esses participantes, os resultados gerais do experimento foram satisfatórios e indicam que a maioria dos participantes foi capaz de seguir o padrão esperado.

Visão 3: Como era de se esperar em um curto período de tempo de explicação do metamodelo, alguns participantes tiveram dificuldades em compreender completamente a utilização dos estereótipos. Por exemplo, cinco participantes apresentaram pequenos erros, como confundir os estereótipos *Plan* e *Plans* (assim como *Goal* e *Goals*), mas esses erros podem ser facilmente corrigidos. Além disso, um participante esqueceu de utilizar o direcionamento nos relacionamentos entre as classes, o que pode levar a interpretações errôneas do diagrama, mas este é um erro compreensível em vista do pouco tempo de estudo.

Outros problemas comuns foram o esquecimento de algum atributo de Percepção, Crença ou Objetivo em diversos casos. É importante lembrar que essa parte da visão é a mais complexa e, portanto, é compreensível que tenham surgido alguns erros. Além disso, foram identificados dois participantes que não utilizaram a classe Plans e outros dois que utilizaram atributos que não estão presentes no metamodelo.

Visão 4: Entre os problemas mais recorrentes, 6 participantes esqueceram de algum atributo de Plan. Em relação aos atributos de *Goal*, 3 participantes não preencheram todos os campos necessários. Entendemos que esses erros podem ter sido resultado de uma compreensão inadequada do conceito de *Goal* e *Plan* ou por falta de atenção na visualização dos parâmetros do metamodelo.

Outros problemas identificados foram: 2 participantes que esqueceram atributos de *Message* e 1 participante que esqueceu atributos de *Action*. Além disso, 2 participantes criaram atributos que não estão presentes no metamodelo para *Action*, o que pode ter sido resultado de falta de compreensão ou familiaridade com esses conceitos.

Além desses, foi identificado 1 participante que trocou o estereótipo "*Goal*" por "*Goals*" e 1 participante que modelou o estereótipo "*Goal*" com os atributos de "*Goals*". Esses erros podem ter sido resultado de falta de atenção, uma vez que as palavras são semelhantes.

Finalmente, foi identificado 1 participante que não criou uma classe para "*Message*" e 1 participante que não criou uma classe para "*Plan*", mesmo a visão sendo focada inteiramente no funcionamento do plano. Esses erros podem ter sido resultado de falta de atenção aos requisitos do modelo.

Por fim, como mencionado anteriormente, o experimento realizado envolveu tarefas complexas que exigem conhecimentos específicos de modelagem. É possível que alguns participantes tenham enfrentado dificuldades na compreensão do cenário utilizado

e, consequentemente, cometido erros ao criar os diagramas. Além disso, pode ter ocorrido falta de atenção ou descuido ao preencher os atributos, resultando em inconsistências nos modelos em comparação ao modelo base e a outros modelos criados pelos próprios participantes. Em conclusão, é importante destacar que a modelagem é uma atividade que requer prática e experiência, e alguns participantes podem ter menos familiaridade com essa atividade do que outros, o que também pode ter impactado na qualidade dos diagramas criados.

QP4: O projeto criado, segue o padrão esperado de 4 modelos com visões específicas de funcionamento?

Resposta: Ao analisar os modelos de diagrama de classes criados pelos participantes, podemos afirmar que a hipótese de que não há diferença significativa entre os projetos dos participantes em relação ao padrão esperado de 4 modelos com visões específicas de funcionamento. é verdadeira. Cada modelo apresentou uma visão clara e específica do funcionamento do SMA, e essas visões foram interconectadas adequadamente.

Cada modelo, por sua vez, descreveu uma perspectiva específica do sistema, abordando aspectos como a comunicação entre os agentes, suas ações e planos. A estrutura geral dos modelos também se encaixa no padrão esperado de 4 modelos, separados por visões. Além disso, as conexões entre os modelos foram bem definidas e mostraram como cada aspecto do sistema se relaciona com os outros. Portanto, é possível concluir que os projetos criados atenderam às expectativas em relação aos modelos de diagrama de classes para SMAs.

5.7 Ameaças a Validade do Estudo

Esta seção tem como objetivo detalhar as ameaças a validade do estudo citadas durante o texto. Desta forma, estas ameaças à validade do estudo que devem ser consideradas, pois podem impactar a interpretação dos resultados obtidos. Abaixo estão as principais ameaças identificadas, juntamente com as medidas tomadas para mitigar esses problemas:

Limitações da amostra e do escopo do experimento: Este estudo foi conduzido como um experimento piloto, com um número limitado de participantes e um escopo específico de análise. Essas limitações podem afetar a generalização dos resultados para uma população mais ampla e para cenários mais complexos de modelagem de SMAs. Mitigação: Apesar das limitações da amostra, foram adotados cuidados na seleção dos participantes para garantir diversidade de experiência e conhecimento. Além disso, as conclusões foram expressas de forma cautelosa, reconhecendo a necessidade de mais experimentos e uma análise mais abrangente para uma compreensão mais completa.

Dificuldades na compreensão do metamodelo e dos requisitos do experimento: O estudo exigiu conhecimentos avançados de modelagem de SMAs, o que pode ter dificultado o entendimento de alguns conceitos do metamodelo e dos requisitos

específicos do experimento. Isso pode ter levado a erros de diagramação e interpretação inadequada dos requisitos. Mitigação: Foram fornecidas explicações e instruções claras sobre o metamodelo e os requisitos do experimento. Os participantes receberam orientações e suporte durante a realização das tarefas, a fim de minimizar as dificuldades de compreensão.

Falta de atenção e desgaste durante a diagramação: Os participantes enfrentaram um desafio de diagramação que exigia atenção e foco. No entanto, a falta de atenção e o desgaste ao longo do processo podem ter afetado a precisão e qualidade dos modelos criados. Embora não tenhamos realizado medidas específicas de mitigação para essa ameaça, entendemos que está ameaça tem uma gravidade baixa. Embora a falta de atenção e o desgaste possam ter afetado os resultados, essa ameaça pode ser minimizada com uma abordagem cuidadosa, pausas adequadas e um ambiente propício para o trabalho.

É importante destacar que, apesar das ameaças à validade identificadas, o estudo piloto forneceu *insights* iniciais e informações relevantes sobre a precisão dos modelos criados, a conformidade com o metamodelo e o padrão esperado de diagramas de classes para SMAs. Essas ameaças podem ser abordadas em estudos futuros mais abrangentes, permitindo uma análise mais robusta e generalizável dos resultados

6 CONSIDERAÇÕES FINAIS

Atualmente, a UML - *Unified Modeling Language* - é a linguagem de modelagem padrão adotada internacionalmente tanto pela indústria de engenharia de software quanto pela academia. Além disso, a maioria dos processos de desenvolvimento de software atuais bem como as ferramentas CASE usam UML para criar seus modelos de software.

Por outro lado, sistemas multiagentes são sistemas de software com características particulares que não são cobertas pela UML padrão. A linguagem é incapaz de representar, por exemplo, agentes, percepções, objetivos, crenças, entre outras características.

Deste modo, a presente dissertação teve como foco a proposta de adaptação e incorporação do Diagrama de Classes da UML à linguagem MASRML, de tal forma que este diagrama possa ser utilizado dentro do contexto de sistemas multiagentes. Para alcançar esse objetivo, foi necessário realizar uma análise das extensões do Diagrama de Classes da UML para o contexto de Sistemas Multiagentes. Essa análise foi realizada através de uma revisão sistemática da literatura em que foram identificados os trabalhos que propuseram adaptações do diagrama de classes para modelar conceitos de Sistemas Multiagentes.

Com base nessa análise, foram selecionados trabalhos que receberam as melhores pontuações segundo a avaliação de qualidade dos estudos aceitos na revisão. Esses trabalhos foram aplicados de forma prática para verificar a sua adequação ao contexto de sistemas multiagentes.

A partir dessas aplicações práticas, foi possível desenvolver uma extensão da linguagem MASRML, adicionando o diagrama de classes UML adaptado para permitir a representação de conceitos particulares de sistemas multiagentes, com foco principal no suporte ao modelo BDI.

A extensão proposta foi avaliada por meio de um experimento de comparação de modelos, no qual foi apontado que a extensão pode ser eficaz na representação de conceitos específicos de sistemas multiagentes. Os resultados obtidos sugerem que o uso da MASRML e a utilização de ferramentas adequadas para a modelagem de sistemas multiagentes é importante, porém é necessário realizar estudos mais aprofundados para entender melhor o impacto da extensão proposta.

A avaliação realizada permitiu chegar a algumas conclusões importantes sobre a proposta de incorporação do Diagrama de Classes da UML à Linguagem MASRML. Primeiramente, os modelos criados apresentaram uma precisão parcial em comparação com a solução desenvolvida pelos proponentes desta extensão. Algumas características do modelo estavam mais próximas da solução esperada pelos proponentes, enquanto outras não apresentaram uma conformidade tão precisa.

Porém, foi possível constatar que os modelos criados foram capazes de descrever com precisão diversos aspectos de agência, permitindo que as características dos agentes

BDI sejam claramente perceptíveis. Além disso, verificou-se que os modelos criados seguem as diretrizes expressas no metamodelo, estando em conformidade com o solicitado. Nesta linha, os projetos criados seguiram o padrão esperado de quatro modelos com visões específicas de funcionamento, interconectadas adequadamente. Desta forma, acreditamos que a proposta de estender a linguagem MASRML, pela adição do diagrama de classes da UML adaptado para o contexto de sistemas multiagentes, pode ser viável e promissora.

A proposta de extensão da linguagem MASRML surge como uma solução para preencher as lacunas encontradas anteriormente nas linguagens analisadas em relação ao suporte ao modelo BDI. É importante destacar que nenhuma das linguagens atuais contempla totalmente os pontos necessários para um suporte adequado ao modelo BDI, especialmente com relação à lógica interna dos agentes. Com a extensão proposta, torna-se possível demonstrar quais percepções afetam determinadas crenças, quais crenças podem influenciar um objetivo, eventualmente tornando-o uma intenção - e quais planos e ações poderão ser disparados caso um objetivo se torne uma intenção.

Concluindo, a proposta de extensão suporta diversos aspectos que anteriormente foram listados como lacunas da área de Sistemas Multiagentes. Entretanto, poderiam ser necessários outros experimentos para melhor avaliar a extensão, considerando o número pequeno de participantes na avaliação e a complexidade da abordagem. Contudo, acreditamos que a extensão seja uma forma válida de modelagem para projetos de Sistemas Multiagentes e pretendemos inserir a proposta dentro do processo REPMAS, um processo para o desenvolvimento de Sistemas Multiagentes, tendo como foco a área de Projeto de Software.

REFERÊNCIAS

ADDOUCHE, N.; ANTOINE, C.; MONTMAIN, J. Methodology for uml modeling and formal verification of real-time systems. In: IEEE. **2006 International Conference on Computational Inteligence for Modelling Control and Automation and International Conference on Intelligent Agents Web Technologies and International Commerce (CIMCA'06)**. [S.l.], 2006. p. 17–17. Citado na página 45.

AGUIAR, E. V. B.; FLÔRES, M. L. P. Objetos de aprendizagem: conceitos básicos. **Objetos de aprendizagem: teoria e prática. Porto Alegre: Evangraf**, p. 12–28, 2014. Citado na página 51.

AL-KADY, M.; BAHGAT, R.; FAHMY, A. A uml heavyweight extension for mas modeling. In: IEEE. **2008 The Eighth International Conference on Quality Software**. [S.l.], 2008. p. 435–440. Citado na página 45.

ALZETTA, F. et al. In-time explainability in multi-agent systems: Challenges, opportunities, and roadmap. In: SPRINGER. **International Workshop on Explainable, Transparent Autonomous Agents and Multi-Agent Systems**. [S.l.], 2020. p. 39–53. Citado na página 32.

ANTONOVA, I.; BATCHKOVA, I. Development of multi-agent control systems using uml/sysml. In: IEEE. **2008 4th International IEEE Conference Intelligent Systems**. [S.l.], 2008. v. 1, p. 6–26. Citado na página 45.

AZAIEZ, S.; HUGET, M.-P.; OQUENDO, F. An approach for multi-agent metamodelling. **Multiagent and Grid Systems**, IOS Press, v. 2, n. 4, p. 435–454, 2006. Citado na página 45.

BADICA, A.; BADICA, C. From formal specification of code mobility to design and implementation: An uml-based mobile agent approach. In: IEEE. **2008 10th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing**. [S.l.], 2008. p. 297–304. Citado 2 vezes nas páginas 45 e 52.

BAKAR, M.; GHOUL, S. A methodology for auml role modeling. In: IEEE. **International Symposium on Innovations in Information and Communications Technology**. [S.l.], 2011. p. 74–81. Citado 2 vezes nas páginas 45 e 55.

BASTOS, R. M.; RIBEIRO, M. B. Masup: An agent-oriented modeling process for information systems. In: SPRINGER. **International Workshop on Software Engineering for Large-Scale Multi-agent Systems**. [S.l.], 2004. p. 19–35. Citado 2 vezes nas páginas 45 e 55.

BAUER, B. Uml class diagrams revisited in the context of agent-based systems. In: SPRINGER. **International Workshop on Agent-Oriented Software Engineering**. [S.l.], 2001. p. 101–118. Citado 2 vezes nas páginas 45 e 55.

BAUER, B.; MULLER, J. P.; ODELL, J. An extension of uml by protocols for multi-agent interaction. In: IEEE. **Proceedings Fourth International Conference on MultiAgent Systems**. [S.l.], 2000. p. 207–214. Citado 2 vezes nas páginas 45 e 55.

BAUER, B.; MÜLLER, J. P.; ODELL, J. Agent uml: A formalism for specifying multiagent software systems. **International journal of software engineering and**

knowledge engineering, World Scientific, v. 11, n. 03, p. 207–230, 2001. Citado 2 vezes nas páginas 45 e 55.

BAUER, B.; ODELL, J. Uml 2.0 and agents: how to build agent-based systems with the new uml standard. **Engineering applications of artificial intelligence**, Elsevier, v. 18, n. 2, p. 141–157, 2005. Citado na página 28.

BELLONI, E.; MARCOS, C. Mam-uml: an uml profile for the modeling of mobile-agent applications. In: IEEE. **XXIV International Conference of the Chilean Computer Science Society**. [S.l.], 2004. p. 3–13. Citado na página 45.

BERGENTI, F.; POGGI, A. Supporting agent-oriented modelling with uml. **International Journal of Software Engineering and Knowledge Engineering**, World Scientific, v. 12, n. 06, p. 605–618, 2002. Citado na página 45.

BIOLCHINI, J. et al. Systematic review in software engineering. **System engineering and computer science department COPPE/UFRJ, Technical Report ES**, v. 679, n. 05, p. 45, 2005. Citado na página 39.

BOES F. MIGEON, F. J. Self-organizing multi-agent systems for the control of complex systems. **Journal of Systems and Software**, Elsevier, v. 134, p. 12–28, 2017. Citado na página 31.

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 na página 33.

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **UML: guia do usuário**. [S.l.]: Elsevier Brasil, 2006. Citado na página 35.

BORDINI, R. H.; HÜBNER, J. F.; WOOLDRIDGE, M. **Programming multi-agent systems in AgentSpeak using Jason**. [S.l.]: John Wiley & Sons, 2007. Citado na página 27.

BOTTI, V. et al. Multi-agent systems. MDPI-Multidisciplinary Digital Publishing Institute, 2019. Citado na página 27.

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 na página 35.

BRATMAN, M. et al. **Intention, plans, and practical reason**. Chicago, USA: Harvard University Press Cambridge, MA, 1987. v. 10. Citado na página 32.

BRESCIANI, P. et al. Tropos: An agent-oriented software development methodology. **Autonomous Agents and Multi-Agent Systems**, Springer, v. 8, n. 3, p. 203–236, 2004. Citado 2 vezes nas páginas 45 e 55.

BRYMAN, A. The research question in social research: what is its role? **International journal of social research methodology**, Taylor & Francis, v. 10, n. 1, p. 5–20, 2007. Citado na página 73.

CAIRE, G. et al. Agent oriented analysis using message/uml. In: SPRINGER. **International Workshop on Agent-Oriented Software Engineering**. [S.l.], 2001. p. 119–135. Citado na página 45.

CALVARESI, D. et al. Explainable multi-agent systems through blockchain technology. In: SPRINGER. **International Workshop on Explainable, Transparent Autonomous Agents and Multi-Agent Systems**. [S.l.], 2019. p. 41–58. Citado na página 27.

CERNUZZI, L. et al. The gaia methodology: Basic concepts and extensions. **Methodologies and software engineering for agent systems: The agent-oriented software engineering handbook**, Springer, p. 69–88, 2004. Citado na página 55.

CERNUZZI, L.; ZAMBONELLI, F. et al. Experiencing auml in the gaia methodology. In: **ICEIS (3)**. [S.l.: s.n.], 2004. p. 283–288. Citado na página 55.

CHOREN, R.; LUCENA, C. The anote modeling language for agent-oriented specification. In: CHOREN, R. et al. (Ed.). **Software Engineering for Multi-Agent Systems III**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. p. 198–212. ISBN 978-3-540-31846-0. Citado 2 vezes nas páginas 33 e 34.

COSSENTINO, M. et al. A notation for modeling jason-like bdi agents. In: IEEE. **2012 Sixth International Conference on Complex, Intelligent, and Software Intensive Systems**. [S.l.], 2012. p. 12–19. Citado na página 45.

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 33.

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 34.

COSSENTINO, M.; SEIDITA, V. Passi: Process for agent societies specification and implementation. In: _____. **Handbook on Agent-Oriented Design Processes**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014. p. 287–329. ISBN 978-3-642-39975-6. Disponível em: <https://doi.org/10.1007/978-3-642-39975-6_10>. Citado 2 vezes nas páginas 33 e 34.

DAVIS, J. P.; BONNELL, R. D. Propositional logic constraint patterns and their use in uml-based conceptual modeling and analysis. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, v. 19, n. 3, p. 427–440, 2007. Citado na página 45.

DENNETT, D. C. **The intentional stance**. [S.l.]: MIT press, 1987. Citado na página 32.

DEPKE, R.; HAUSMANN, J. H.; HECKEL, R. Design of an agent-oriented modeling language based on graph transformation. In: SPRINGER. **International Workshop on Applications of Graph Transformations with Industrial Relevance**. [S.l.], 2003. p. 106–119. Citado na página 45.

DINSOREANU, M.; SALOMIE, I.; PUSZTAI, K. On the design of agent-based systems using uml and extensions. In: IEEE. **ITI 2002. Proceedings of the 24th International Conference on Information Technology Interfaces (IEEE Cat. No. 02EX534)**. [S.l.], 2002. p. 205–210. Citado na página 45.

EHRLER, L.; CRANEFIELD, S. Executing agent uml diagrams. In: **Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 2**. [S.l.: s.n.], 2004. p. 906–913. Citado 2 vezes nas páginas 45 e 55.

FUENTES, R.; GÓMEZ-SANZ, J. J.; PAVÓN, J. Integrating agent-oriented methodologies with uml-at. In: **Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems**. [S.l.: s.n.], 2006. p. 1303–1310. Citado na página 45.

GAN, K. et al. Enforcing social semantic in fipa-acl using spin. In: _____. Singapore: Springer, 2020. p. 3–13. ISBN 978-981-13-8678-7. Citado na página 27.

GARCíA-OJEDA, J.; DELOACH, S. The o-mase methodology. In: _____. Berlin, Heidelberg: Springer, Berlin, Heidelberg, 2014. p. 253–285. ISBN 978-3-642-39974-9. Citado na página 33.

GARIJO, F. J.; GOMEZ-SANZ, J. J.; MASSONET, P. The message methodology for agent-oriented analysis and design. In: **Agent-oriented methodologies**. [S.l.]: IGI Global, 2005. p. 203–235. Citado na página 45.

GETIR, S.; CHALLENGER, M.; KARDAS, G. The formal semantics of a domain-specific modeling language for semantic web enabled multi-agent systems. **International Journal of Cooperative Information Systems**, World Scientific, v. 23, n. 03, p. 1450005, 2014. Citado 4 vezes nas páginas 32, 45, 58 e 64.

GOGOLLA, M.; HENDERSON-SELLERS, B. Analysis of uml stereotypes within the uml metamodel. In: SPRINGER. **UML**. [S.l.], 2002. v. 2, p. 84–99. Citado na página 37.

GONÇALVES, E. J. T. et al. Extending mas-ml to model proactive and reactive sotware agents. In: **ICEIS (2)**. [S.l.: s.n.], 2010. p. 75–84. Citado 2 vezes nas páginas 45 e 52.

GONÇALVES, E. J. T. et al. Mas-ml 2.0: Supporting the modelling of multi-agent systems with different agent architectures. **Journal of Systems and Software**, Elsevier, v. 108, p. 77–109, 2015. Citado 2 vezes nas páginas 45 e 52.

GREEN, H. **Business Modeling with UML–Business Patterns at Work**. [S.l.]: Oxford University Press, 2000. Citado na página 35.

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, p. 21, 2020. Citado 3 vezes nas páginas 34, 35 e 45.

GUEDES, G. T. Uml 2. **Uma Abordagem Prática", São Paulo, Novatec**, p. 32, 2009. Citado 2 vezes nas páginas 28 e 52.

GUEDES, G. T. **UML 2-Uma abordagem prática**. [S.l.]: Novatec Editora, 2018. Citado 4 vezes nas páginas 28, 35, 36 e 37.

GUEDES, G. T. A. **Um metamodelo UML para a modelagem de requisitos em projetos de sistemas multiagentes**. Tese (Doutorado) — Universidade Federal do Rio Grande do Sul (UFRGS), 2012. Citado na página 51.

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 28.

GUEDES, G. T. A.; VICARI, R. M. Specific uml-derived languages for modeling multi-agent systems. In: **Handbook on Artificial Intelligence-Empowered Applied Software Engineering**. [S.l.]: Springer, 2022. p. 159–223. Citado 3 vezes nas páginas 40, 49 e 53.

HACHICHA, H.; LOUKIL, A.; GHEDIRA, K. Ma-uml: a conceptual approach for mobile agents' modelling. **International Journal of Agent-Oriented Software Engineering**, Inderscience Publishers, v. 3, n. 2-3, p. 277–305, 2009. Citado na página 45.

HAHN, C.; JACOBI, S.; RABER, D. Enhancing the interoperability between multiagent systems and service-oriented architectures through a model-driven approach. In: SPRINGER. **German Conference on Multiagent System Technologies**. [S.l.], 2010. p. 88–99. Citado na página 45.

HAJDUK, M.; SUKOP, M.; HAUN, M. **Cognitive Multi-agent Systems**. USA: Springer, 2019. Citado 2 vezes nas páginas 27 e 31.

HERCHI, H.; ABDESSALEM, W. B. From user requirements to uml class diagram. **arXiv preprint arXiv:1211.0713**, 2012. Citado na página 35.

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 27.

HUGET, M.-P. Agent uml class diagrams revisited. In: SPRINGER. **Net. ObjectDays: International Conference on Object-Oriented and Internet-Based Technologies, Concepts, and Applications for a Networked World**. [S.l.], 2002. p. 49–60. Citado 2 vezes nas páginas 45 e 55.

HUGET, M.-P. Agent uml notation for multiagent system design. **IEEE Internet Computing**, IEEE, v. 8, n. 4, p. 63–71, 2004. Citado 2 vezes nas páginas 45 e 55.

HUGET, M.-P.; ODELL, J. Representing agent interaction protocols with agent uml. In: SPRINGER. **International Workshop on Agent-Oriented Software Engineering**. [S.l.], 2004. p. 16–30. Citado 2 vezes nas páginas 45 e 55.

JEUSFELD, M. A. Metamodel. In: _____. **Encyclopedia of Database Systems**. Boston, MA: Springer US, 2009. p. 1727–1730. ISBN 978-0-387-39940-9. Disponível em: <https://doi.org/10.1007/978-0-387-39940-9_898>. Citado na página 37.

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**. [S.l.: s.n.], 2002. p. 3–10. Citado na página 55.

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.

KOSIUCZENKO, P. Sequence diagrams for mobility. In: SPRINGER. **International Conference on Conceptual Modeling**. [S.l.], 2002. p. 147–158. Citado na página 45.

LAOUADI, M. A.; MOKHATI, F.; SERIDI-BOUCHELAGHEM, H. A novel formal specification approach for real time multi-agent system functional requirements. In: SPRINGER. **German Conference on Multiagent System Technologies**. [S.l.], 2010. p. 15–27. Citado 2 vezes nas páginas 45 e 55.

LAOUADI, M. A.; MOKHATI, F.; SERIDI-BOUCHELAGHEM, H. Towards an organizational model for real time multi-agent system specification. In: IEEE. **2013 Science and Information Conference**. [S.l.], 2013. p. 577–584. Citado 2 vezes nas páginas 45 e 55.

LAOUADI, M. A.; MOKHATI, F.; SERIDI, H. A novel organizational model for real time mas: Towards a formal specification. In: **Intelligent Systems for Science and Information**. [S.l.]: Springer, 2014. p. 171–180. Citado 2 vezes nas páginas 45 e 55.

LARSEN, J. B. Agent programming languages and logics in agent-based simulation. In: **Modern approaches for intelligent information and database systems**. [S.l.]: Springer, 2018. p. 517–526. Citado na página 32.

LEE, C.-H. L.; LIU, A. A method for agent-based system requirements analysis. In: IEEE. **Fourth International Symposium on Multimedia Software Engineering, 2002. Proceedings.** [S.l.], 2002. p. 214–221. Citado na página 45.

LIU, J.-X. et al. The aml approach to modeling dynamic organization of combat team. In: IEEE. **2010 2nd International Workshop on Intelligent Systems and Applications**. [S.l.], 2010. p. 1–4. Citado na página 45.

LIU L. ZHENG, T. Z. G. Cooperative planning process modeling on daily dispatching plan in railway bureau based on multi-agent. In: **2010 International Conference on Intelligent System Design and Engineering Application**. [S.l.: s.n.], 2011. v. 2, p. 616–621. Citado na página 31.

MURRAY, J.; STOLZENBURG, F. Hybrid state machines with timed synchronization for multi-robot system specification. In: IEEE. **2005 portuguese conference on artificial intelligence**. [S.l.], 2005. p. 236–241. Citado na página 45.

MYLOPOULOS, J.; KOLP, M.; CASTRO, J. Uml for agent-oriented software development: The tropos proposal. In: SPRINGER. **International Conference on the Unified Modeling Language**. [S.l.], 2001. p. 422–441. Citado na página 45.

NITTO, E. D. et al. Deriving executable process descriptions from uml. In: **Proceedings of the 24th International Conference on Software Engineering**. [S.l.: s.n.], 2002. p. 155–165. Citado na página 35.

OMG. **OMG Unified Modeling Language (OMG UML), Superstructure, Version 2.4.1**. 2011. Disponível em: <http://www.omg.org/spec/UML/2.4.1>. Citado na página 35.

PADGHAM, L.; WINIKOFF, M. Prometheus: A methodology for developing intelligent agents. In: SPRINGER. **International Workshop on Agent-Oriented Software Engineering**. [S.l.], 2002. p. 174–185. Citado na página 55.

PAPASIMEON, M.; HEINZE, C. Extending the uml for designing jack agents. In: IEEE. **Proceedings 2001 Australian Software Engineering Conference**. [S.l.], 2001. p. 89–97. Citado na página 45.

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**. San Francisco, CA, USA: ICMAS, 1995. p. 312–319. Citado na página 32.

RAO, A. S.; GEORGEFF, M. P. et al. Bdi agents: from theory to practice. In: **ICMAS**. [S.l.: s.n.], 1995. v. 95, p. 312–319. Citado na página 31.

RUSSELL, S. J.; NORVIG, P. **Artificial intelligence: a modern approach**. Upper Saddle River: Malaysia; Pearson Education Limited„ 2016. Citado na página 33.

SADOWSKA, M. **Creating and validating UML class diagrams with the use of domain ontologies expressed in OWL 2**. Tese (Doutorado) — Doctoral Thesis, Faculty of Computer Science and Management, Wroclaw . . . , 2020. Citado na página 28.

SILVA, C. et al. Improving multi-agent architectural design. In: SPRINGER. **International Workshop on Software Engineering for Large-Scale Multi-agent Systems**. [S.l.], 2006. p. 165–184. Citado 2 vezes nas páginas 45 e 56.

SILVA, C. T. et al. Towards a standardized description and a systematic use of social patterns. In: **CIbSE**. [S.l.: s.n.], 2007. p. 195–208. Citado 2 vezes nas páginas 45 e 56.

SILVA, L. D.; MENEGUZZI, F. R.; LOGAN, B. Bdi agent architectures: A survey. In: **Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI), 2020, Japão.** [S.l.: s.n.], 2020. Citado na página 33.

SILVA, V. T. D.; CHOREN, R.; LUCENA, C. J. P. D. Mas-ml: A multiagent system modelling language. **Int. J. Agent-Oriented Softw. Eng.**, Inderscience Publishers, Geneva 15, CHE, v. 2, n. 4, p. 382–421, aug 2008. ISSN 1746-1375. Disponível em: <https://doi.org/10.1504/IJAOSE.2008.020138>. Citado na página 53.

SILVA, V. T. D.; LUCENA, C. J. D. From a conceptual framework for agents and objects to a multi-agent system modeling language. **Autonomous Agents and Multi-Agent Systems**, Springer, v. 9, n. 1, p. 145–189, 2004. Citado 2 vezes nas páginas 45 e 52.

SILVA, V. T. D.; LUCENA, C. J. de. Mas-ml: a multi-agent system modeling language. In: **Companion of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications**. [S.l.: s.n.], 2003. p. 126–127. Citado na página 52.

SILVA, V. T. da; NOYA, R. C.; LUCENA, C. J. de. Using the uml 2.0 activity diagram to model agent plans and actions. In: **Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems**. [S.l.: s.n.], 2005. p. 594–600. Citado 2 vezes nas páginas 45 e 52.

SILVA, V. Torres da; CHOREN, R.; LUCENA, C. J. D. Modeling mas properties with mas-ml dynamic diagrams. In: SPRINGER. **International Bi-Conference Workshop on Agent-Oriented Information Systems**. [S.l.], 2006. p. 1–18. Citado 2 vezes nas páginas 45 e 52.

SINGH, D.; PADGHAM, L.; LOGAN, B. Integrating bdi agents with agent-based simulation platforms. **Autonomous Agents and Multi-Agent Systems**, Springer, v. 30, n. 6, p. 1050–1071, 2016. Citado na página 32.

SOUZA, I.; GUEDES, G.; MENDONÇA, G. D. Repmas: A requirements engineering process for multiagent systems: An application example. In: **Proceedings of the15th International Conference on Agents and Artificial Intelligence (ICAART)**. [S.l.: s.n.], 2023. Citado na página 35.

SPANOUDAKIS, N.; MORAITIS, P. An agent modeling language implementing protocols through capabilities. In: IEEE. **2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology**. [S.l.], 2008. v. 2, p. 578–582. Citado na página 45.

SUBBURAJ, V. H.; URBAN, J. E. Specifying security requirements in multi-agent systems using the descartes-agent specification language and auml. In: **Information technology for management: Emerging research and applications**. [S.l.]: Springer, 2018. p. 93–111. Citado 2 vezes nas páginas 45 e 55.

TAILLANDIER, P. et al. A bdi agent architecture for the gama modeling and simulation platform. In: SPRINGER. **Multi-Agent Based Simulation XVII: International Workshop, MABS 2016, Singapore, Singapore, May 10, 2016, Revised Selected Papers 17**. [S.l.], 2017. p. 3–23. Citado na página 33.

THUNG, F. et al. Condensing class diagrams by analyzing design and network metrics using optimistic classification. In: **Proceedings of the 22nd International Conference on Program Comprehension**. [S.l.: s.n.], 2014. p. 110–121. Citado na página 28.

TRENCANSKY, I.; CERVENKA, R. Agent modeling language (aml): A comprehensive approach to modeling mas. **Informatica**, v. 29, n. 4, 2005. Citado na página 45.

TRENCANSKY, I.; CERVENKA, R.; GREENWOOD, D. Applying a uml-based agent modeling language to the autonomic computing domain. In: **Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications**. [S.l.: s.n.], 2006. p. 521–529. Citado na página 45.

TURNER, C. K. A principle of intentionality. **Frontiers in Psychology**, Frontiers Media SA, v. 8, p. 137, 2017. Citado na página 32.

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 na página 31.

VIRUEL, M. L. R. Requirements modeling for multi-agent systems. In: _____. **Multi-Agent Systems - Modeling, Control, Programming, Simulations and Applications**. Asunción, Paraguay: InTech, 2011. cap. 1, p. 3–22. Citado na página 27.

WAGNER, G. The agent–object-relationship metamodel: towards a unified view of state and behavior. **Information Systems**, Elsevier, v. 28, n. 5, p. 475–504, 2003. Citado na página 45.

WIENBERG, A.; MATTHES, F.; BOGER, M. Modeling dynamic software components in uml. In: SPRINGER. **International Conference on the Unified Modeling Language**. [S.l.], 1999. p. 204–219. Citado na página 45.

Wilmann, D.; Sterling, L. Guiding agent-oriented requirements elicitation: Homer. In: **Fifth International Conference on Quality Software (QSIC'05)**. Melbourne, VIC, Australia: IEEE, 2005. p. 419–424. Citado na página 35.

WINIKOFF, M. Towards making agent uml practical: A textual notation and a tool. In: IEEE. **Fifth International Conference on Quality Software (QSIC'05)**. [S.l.], 2005. p. 401–406. Citado 2 vezes nas páginas 45 e 55.

WOHLIN, C. et al. **Experimentation in software engineering**. [S.l.]: Springer Science & Business Media, 2012. Citado na página 71.

WOOLDRIDGE, M. **An introduction to multiagent systems**. USA: John Wiley & Sons, 2009. Citado na página 31.

XIANG, Z.; ZHI-QING, S. Asm semantic modeling and checking for sequence diagram. In: IEEE. **2009 Fifth International Conference on Natural Computation**. [S.l.], 2009. v. 5, p. 527–530. Citado na página 45.

XIAO, L. et al. Adaptive agent model: an agent interaction and computation model. In: IEEE. **31st Annual International Computer Software and Applications Conference (COMPSAC 2007)**. [S.l.], 2007. v. 2, p. 153–158. Citado na página 45.

YIM, H. S.; LEE, H.; PARK, S. J. Adam: architecture-driven multi-agent systems development methods. **WSEAS Transactions on Systems**, Citeseer, v. 4, n. 10, p. 1762–1767, 2005. Citado na página 45.

ZEID, A. A uml profile for agent-based development. In: SPRINGER. **International Symposium on Metainformatics**. [S.l.], 2002. p. 161–170. Citado na página 45.

Anexos

ANEXO A – INSTRUÇÕES DO EXPERIMENTO

A.1 Orientações

- Realize a leitura do Sistema descrito abaixo.
- Através de uma ferramenta de modelagem UML faça os diagramas necessários para satisfazer as 4 visões explicadas anteriormente.
  - A primeira e a segunda visão devem expressar todo o sistema.
  - A terceira visão deve detalhar apenas o objetivo referente ao cenário de “Garantir a segurança dos pedestres e dos veiculos”.
  - A quarta visão deve detalhar apenas o plano referente ao cenário de “Sinalização para veículos"(em caso de dúvida, siga à diagramação para o plano citado no objetivo que foi detalhado na visão anterior.)
- O participante poderá visualizar as visões do metamodelo na parte inferior deste documento.
- Em caso de dúvida quanto ao funcionamento do experimento, o participante poderá perguntar para o pesquisador responsável.

A.2 Sistema multi-agente de controle de trânsito

O sistema multi-agente de controle de trânsito tem como objetivo garantir a segurança e a fluidez do tráfego nas vias públicas. O agente de trânsito é responsável por monitorar o fluxo de veículos, bem como sinalizar e controlar o tráfego em cruzamentos e semáforos.

Além disso, o sistema também conta com veículos autônomos que são capazes de se comunicar com o agente de trânsito para receber orientações e ajustar a velocidade conforme as condições do tráfego.

Quando um grande número de pedestres precisa atravessar uma via, o agente de trânsito aciona o controle de tráfego para travessia de pedestres, garantindo a segurança dos pedestres e regulando o tráfego dos veículos.

Assim, o sistema multi-agente de controle de trânsito é capaz de integrar diversos agentes, veículos e pedestres em um ambiente colaborativo e seguro, contribuindo para a redução de acidentes de trânsito e a melhoria da qualidade de vida nas cidades.

A.2.1 Agente de trânsito controlando o fluxo de veículos

- Agente de trânsito acredita que é necessário agir para evitar congestionamento.
- O desejo do Agente de trânsito é garantir a segurança e a fluidez do trânsito.

- Agente de trânsito percebe que o fluxo de veículos está aumentando através da sua observação do tráfego na rua.
- O plano do Agente de trânsito é tomar medidas para controlar o tráfego e evitar congestionamentos.
- As ações do Agente de trânsito podem ser (I) sinalizar para que os veículos reduzam a velocidade, (II) bloquear o acesso de veículos em uma determinada via, (III) redirecionar o tráfego para evitar congestionamentos ou (IV) utilizar equipamentos de som ou sinalizadores luminosos para alertar os veículos.

A.2.2 Controle de tráfego para travessia de pedestres

- Agente de trânsito acredita que é necessário organizar o fluxo de veículos.
- O desejo do Agente de trânsito é garantir a segurança dos pedestres e dos veículos.
- Agente de trânsito percebe que há pedestres esperando para atravessar a faixa de pedestres através da sua observação do tráfego na rua.
- O plano do Agente de trânsito é sinalizar para os veículos reduzirem a velocidade e aguardarem a passagem dos pedestres na faixa de pedestres.
- As ações do Agente de trânsito são executadas ao enviar uma mensagem aos veículos autônomos presentes na via e aplicar a sinalização visual e sonora para os motoristas.
- O veículo autônomo acredita que deve seguir as regras de trânsito e reduzir a velocidade ao se aproximar da faixa de pedestres.
- O desejo do veículo autônomo é evitar acidentes com pedestres.
- O veículo autônomo percebe a presença de pedestre na faixa de pedestres através da sua observação do tráfego na rua ou percebe o envio de mensagem do agente de trânsito notificando a presença de pedestres.
- O plano do veículo autônomo é aguardar a passagem dos pedestres para isso, primeiro ele deve acionar o plano de reduzir velocidade.
- A ação do veículo autônomo para reduzir a velocidade é acionar o freio e neste caso parar o veículo para permitir que o pedestre atravesse a faixa de pedestres.
- Pedestre acredita que deve atravessar a faixa de pedestres quando ela estiver livre de veículos.
- O desejo do Pedestre é atravessar a rua em segurança.

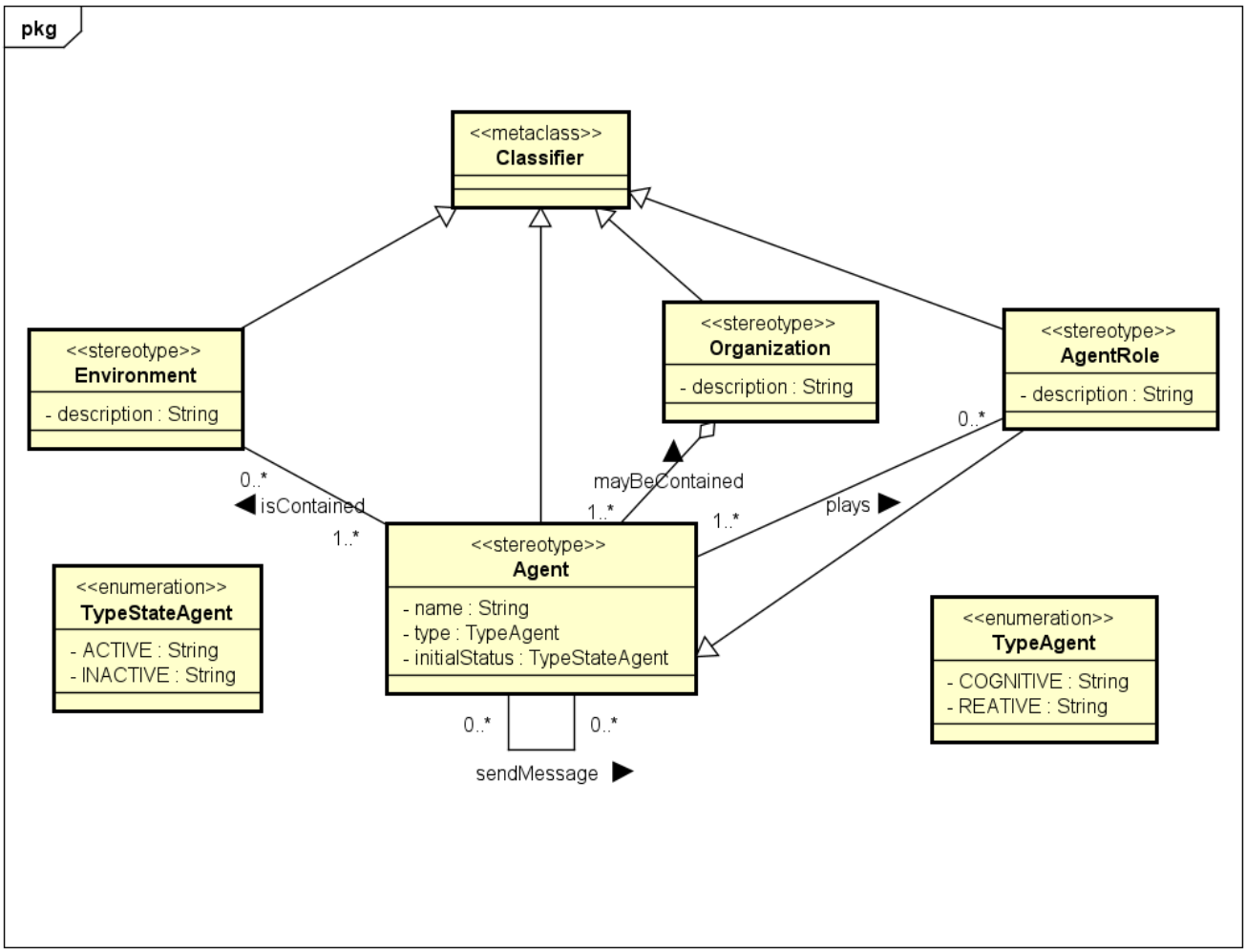
- Pedestre percebe que veículo autônomo reduziu a velocidade e parou o veículo através da sua observação do tráfego na rua.
- O plano de Pedestre é atravessar a faixa de pedestres quando ela estiver livre de veículos.
- As ações de Pedestre são executar o seu plano, por exemplo, atravessar a faixa de pedestres quando ela estiver livre de veículos.

A.2.3 Redução de velocidade do veículo

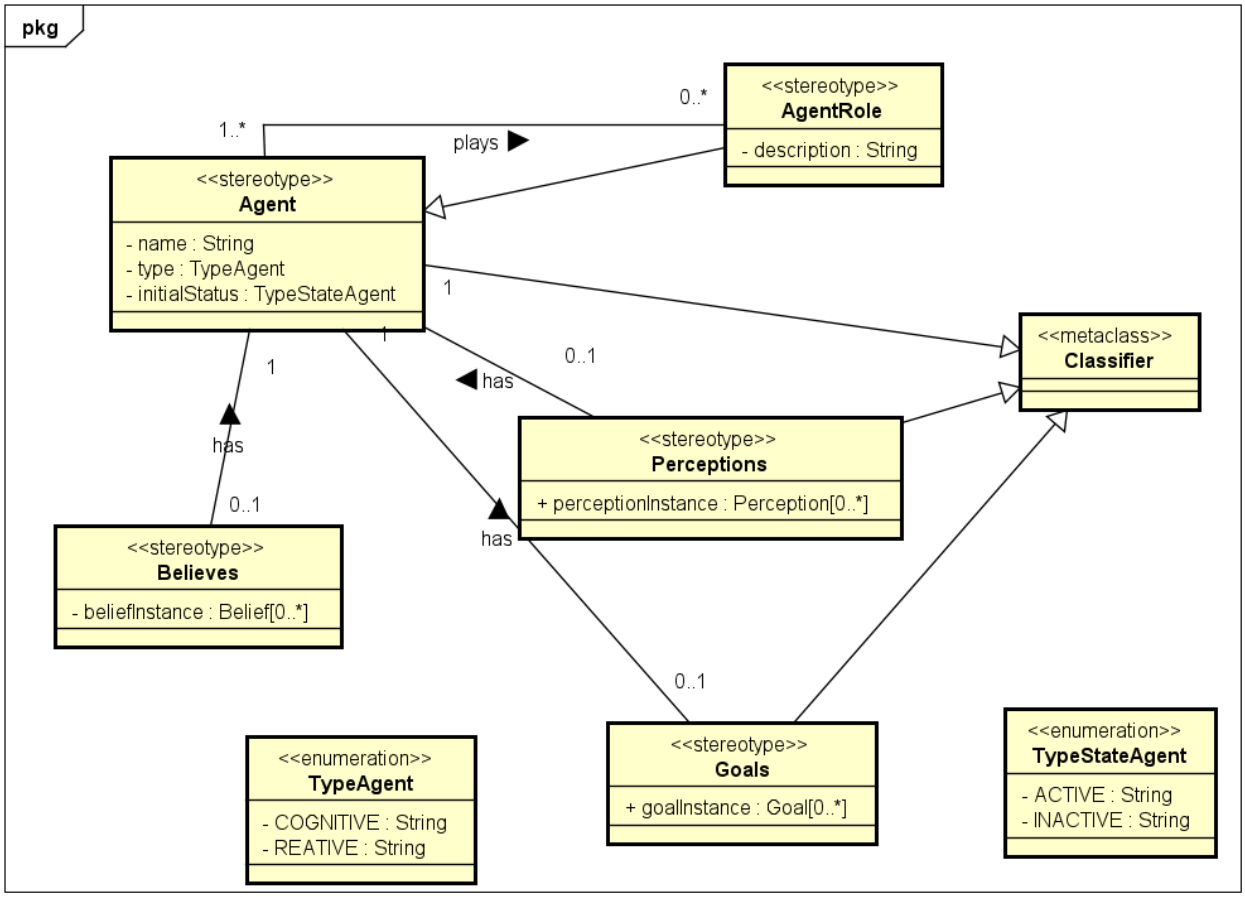
- O veículo autônomo deseja chegar ao seu destino com segurança e rapidez. Acredita que pode dirigir em alta velocidade sem colocar outros motoristas em perigo.
- Agente de trânsito percebe que o veículo autônomo está dirigindo em alta velocidade e sinaliza para o veículo reduzir a velocidade.
- O veículo autônomo percebe o sinal do Agente de trânsito e entende que precisa reduzir a velocidade.
- O veículo autônomo tem o plano de reduzir a velocidade para atender à ordem do Agente de trânsito.
- A ação do veículo deve ser acionar o freio.

A.3 Visões

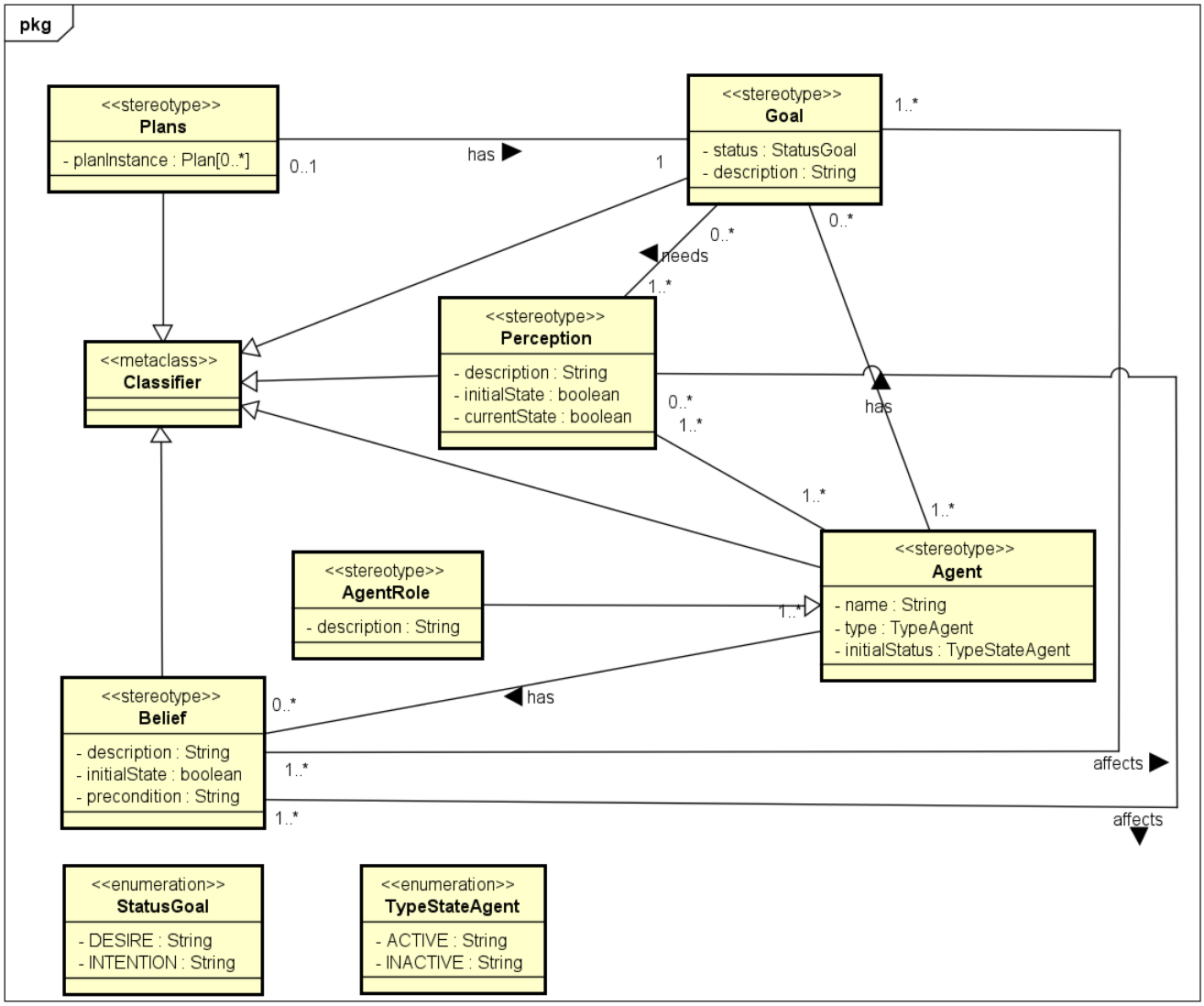
A.3.1 Primeira Visão:



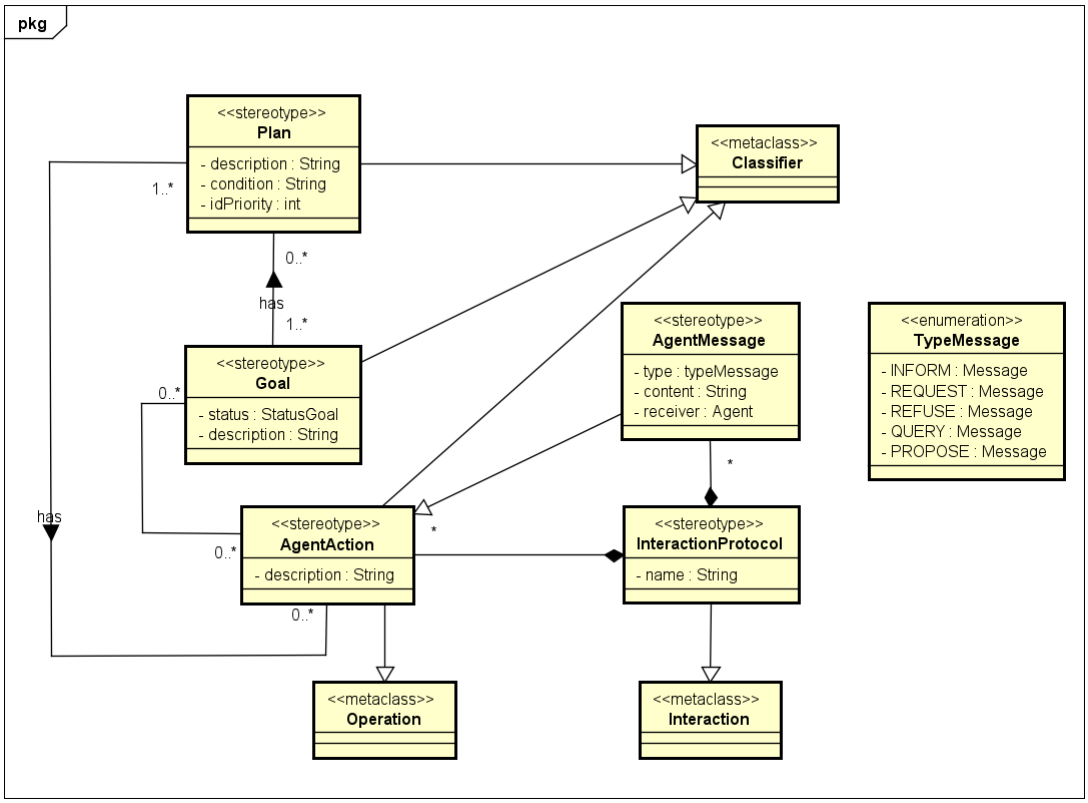
A.3.2 Segunda Visão:



A.3.3 Terceira Visão:



A.3.4 Quarta Visão:



ANEXO B – NÍVEL TÉCNICO DOS PARTICIPANTES DO EXPERIMENTO.

Formulário para determinar o nível técnico do participante

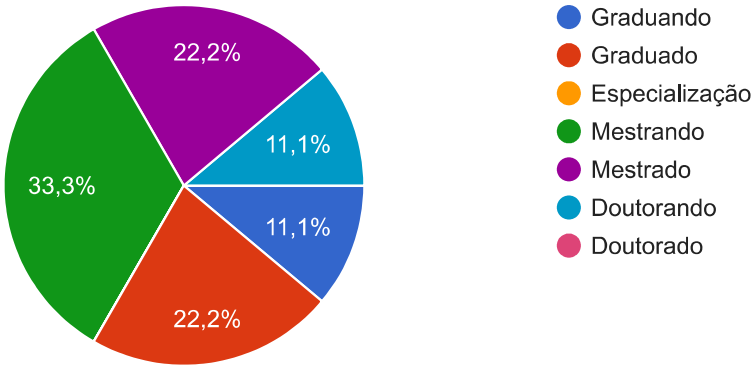
9 respostas

Publicar análise

Qual seu nível de formação?

Copiar

9 respostas



(Se respondeu "Graduando"). Atualmente qual o curso de graduação que está frequentando?

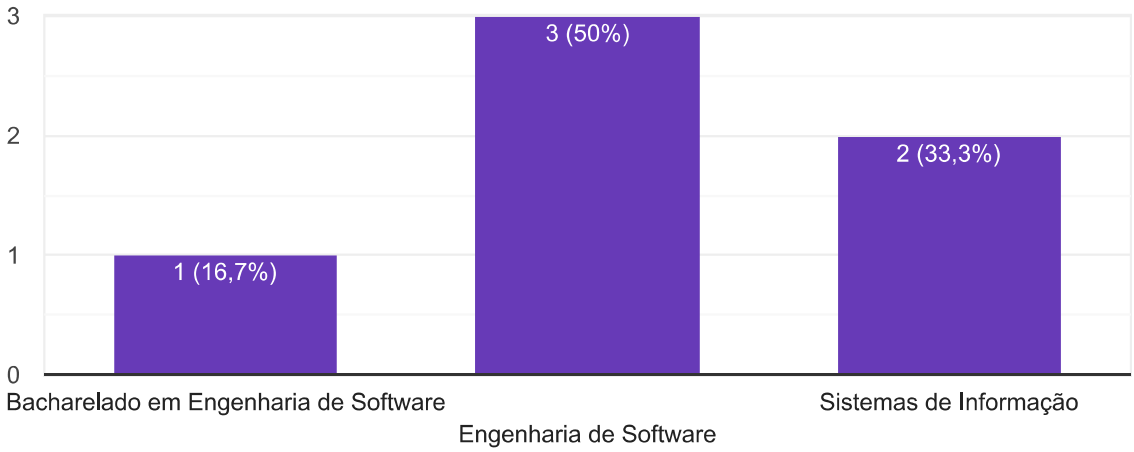
1 resposta

Engenharia de software

Qual o curso de graduação que concluiu?

Copiar

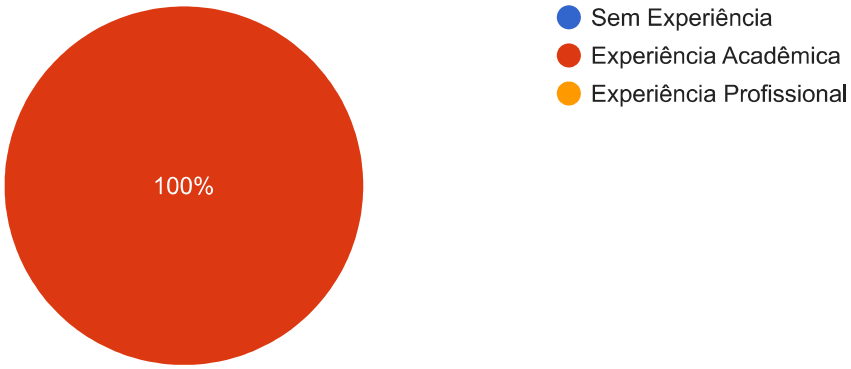
6 respostas



Qual o seu nível de conhecimento em sistemas multiagentes?

Copiar

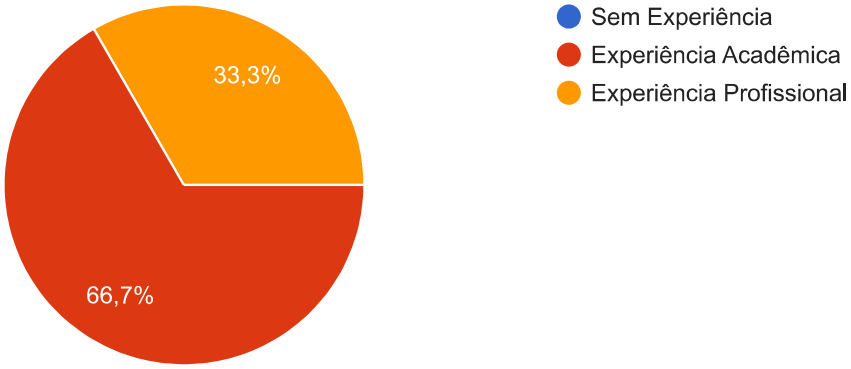
9 respostas



Qual o seu nível de conhecimento em modelagem UML?

Copiar

9 respostas



ÍNDICE