UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS QUIXADÁ

CURSO DE GRADUAÇÃO EM SISTEMAS DE INFORMAÇÃO

RENDLEY ARNOU XAVIER

PSMAR: UM PROCESSO PARA DESENVOLVIMENTO DE SISTEMAS MULTIAGENTES COM AGENTES RACIONAIS

QUIXADÁ

2022

RENDLEY ARNOU XAVIER

PSMAR: UM PROCESSO PARA DESENVOLVIMENTO DE SISTEMAS MULTIAGENTES COM AGENTES RACIONAIS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Sistemas de Informação do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Sistemas de Informação.

Orientador: Prof. Dr. Enyo José Tavares Gonçalves.

QUIXADÁ

2022

Dados Internacionais de Catalogação na Publicação

Universidade Federal do Ceará

Sistema de Bibliotecas

Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo autor

---

X23p Xavier, Rendley Arnou.

PSMAR : um processo para desenvolvimento de sistemas multiagentes com agentes racionais / Rendley Arnou Xavier. – 2022.

94 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Sistemas de Informação, Quixadá, 2022.

Orientação: Prof. Dr. Enyo José Tavares Gonçalves.

1. Engenharia de Software. 2. Inteligência Artificial. 3. Multiagent systems. I. Título.

CDD 005

RENDLEY ARNOU XAVIER

PSMAR: UM PROCESSO PARA DESENVOLVIMENTO DE SISTEMAS MULTIAGENTES COM AGENTES RACIONAIS

Trabalho de Conclusão de Curso apresentado ao Curso de Sistemas de Informação do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Sistemas de Informação.

Aprovado em: _____/_____/_____.

BANCA EXAMINADORA

_____
Prof. Dr. Enyo José Tavares Gonçalves (Orientador)
Universidade Federal do Ceará (UFC)

_____
Prof. Dr. Marcos Antônio de Oliveira
Universidade Federal do Ceará (UFC)

_____
Prof. Me. Camilo Camilo Almendra
Universidade Federal do Ceará (UFC)

A Deus.

Aos meus pais, Maria Luzia e José Auri. Meus irmãos, família e Amigos.

AGRADECIMENTOS

Inicialmente a Deus, que sempre esteve ao meu lado. Obrigado pela força, ânimo, coragem, amor e saúde que me forneceu ao longo deste processo complicado e desgastante ao ponto de superar as dificuldades, pois sem sua graça não seria capaz de alcançar a conclusão deste trabalho e continuar acreditando no meu sonho e lutar por alcançar aquilo que acredito.

À CAPES, pelo apoio financeiro com a manutenção da bolsa de auxílio.

A minha família, em especial meus pais, Maria Luzia Eduardo Arnou e José Auri Xavier e meus irmãos Renan Arnou Xavier e Renê Arnou Xavier pelo apoio, suporte, carinho, conselhos e amor incondicional.

Aos meus parentes, em especial José Sinval, Maria da Conceição (in memoriam), Elizabete Correia, Glória Arnou, Amanda Quaresma, Ângela Arnou, Fabiana Correia, Andreia Arnou, Carlos Arnou, Salete Arnou, Haroldo Arnou e a todos que de alguma forma contribuíram com todo apoio, amparo e carinho durante toda minha caminhada.

A Jailson Bastos, Vinicius Alves, Yan Lima, Renan Gomes, Pedro Venicius, Luanderson Aires e Ronier Lima, pela amizade verdadeira e companheirismo, e todas as pessoas que contribuíram de forma positiva para minhas experiências durante toda a graduação.

A Elida, pela paciência e por ter caminhado ao meu lado e acreditado nos meus sonhos.

Aos meus amigos pela amizade e apoio.

Ao Prof. Dr. Enyo José Tavares Gonçalves, por ter aceitado me conduzir nesta caminhada, pela paciência e pela excelente orientação.

A todos os professores que tive o prazer de conhecer e que contribuíram para o meu conhecimento.

Aos professores participantes da banca examinadora Prof. Dr. Marcos Antônio de Oliveira e Prof. Me. Camilo Camilo Almendra pelo tempo, pelas valiosas colaborações e sugestões.

Aos professores e alunos entrevistados, pelo tempo concedido nas entrevistas.

A quem não mencionei e aos colegas da turma de graduação, pelas reflexões, críticas e sugestões recebidas.

“Estou em encantador estado de confusão.” (Ada Lovelace).

RESUMO

Este trabalho propõe a criação de um processo para apoiar o desenvolvimento de software para sistemas multiagentes com agentes racionais integrando técnicas já existentes para desenvolvimento de sistemas multiagentes com agentes a nível de requisitos, projeto e implementação. Para isso, foi realizado o estudo de modelos de processo de software, onde cada modelo de processo representa um processo sob determinada perspectiva, e dessa forma, viabilizar a criação do modelo proposto fundamentando nas variações desses processos e então estabelecer seus principais estágios que definem as atividades fundamentais de desenvolvimento. Como resultado, foi desenvolvido o PSMAR: Um Processo para Desenvolvimento de Sistemas Multiagentes com Agentes Racionais, composto por 8 fases. O presente trabalho proposto foi validado baseado por meio de uma abordagem com pesquisadores e desenvolvedores que atuam com o desenvolvimento de sistemas multiagentes, onde o objetivo da avaliação foi identificar os pontos de vista sobre o PSMAR e com isso, contribuir como um processo de referência para orientar outros processos para desenvolvimento.

Palavras-chave: engenharia de software; inteligência artificial; sistemas multiagentes.

ABSTRACT

This work proposes the creation of a process to support the development of software for multi-agent systems with rational agents, integrating existing techniques for the development of multi-agent systems with agents at the level of requirements, design and implementation. For this, the study of software process models was carried out, where each process model represents a process from a certain perspective, and in this way, enable the creation of the proposed model based on the variations of these processes and then establish its main stages that define the core development activities. As a result, PSMAR was developed: A Process for the Development of Multi-agent Systems with Rational Agents, composed of 8 phases. The present proposed work was validated based on an approach with researchers and developers who work with the development of multi-agent systems, where the objective of the evaluation was to identify the points of view about PSMAR and with that, to contribute as a reference process for guide other processes for development.

Keywords: software engineering; artificial intelligence; multiagent systems.

LISTA DE FIGURAS

Figura 1 - Visão geral de um agente 22
Figura 2 - Visão geral da ferramenta piStar4RationalAgents 25
Figura 3 - Modelagem do modelo SD do iStar para o MOODLE usando a nova extensão (GONÇALVES, et al. 2019) 27
Figura 4 - Modelagem do modelo SR do iStar para o MOODLE usando a nova extensão (GONÇALVES et al. 2019) 29
Figura 5 - Metamodelo da extensão do MAS-ML 2.0 31
Figura 6 - Representação dos agentes com diagrama estáticos 32
Figura 7 - Representação de papéis para agente reflexivo simples e agente reativo baseado em conhecimento 32
Figura 8 - Environment template (Lopes, et al. 2018) 35
Figura 9 - Fluxo principal do PSMAR 40
Figura 10 - Levantar e especificar requisitos do sistema multiagente 41
Figura 11 - Detalhar requisitos 43
Figura 12 - Projetar sistema multiagente com MAS-ML 2.0 43
Figura 13 - Desenvolver sistema multiagente com JAMDER 45
Figura 14 - Testar sistema multiagente 46
Figura 15 - Diagrama de dependência estratégica do MOODLE 49
Figura 16 - Diagrama de raciocínio estratégico do MOODLE 50
Figura 17 - Diagrama de papéis de MAS-ML 2.0 do MOODLE 51
Figura 18 - Diagrama de classes de MAS-ML 2.0 do MOODLE 52
Figura 19 - Classe MoodleEnv desenvolvida com o framework JAMDER 53
Figura 20 - Nível de experiência dos participantes com desenvolvimento de sistemas multiagentes 56
Figura 21 - Quantidade de participantes que conhecem outro processo para desenvolvimento de sistemas multiagentes com agentes racionais 56
Figura 22 - É necessária a proposição de um processo para conduzir o desenvolvimento de SMAs com agentes racionais 57
Figura 23 - A especificação textual de requisitos do PSMAR ajuda a estruturar bem o SMA nas fases iniciais do desenvolvimento 57
Figura 24 - O gerenciamento das iterações do PSMAR é relevante para a execução do

processo 58
Figura 25 - Regras de mapeamento entre modelagem de requisitos com iStar4RationalAgents e MAS-ML 2.0 são relevantes para o desenvolvimento de SMAs com agentes racionais 58
Figura 26 - Os tutoriais existentes no PSMAR são úteis para facilitar a configuração para modelar com a ferramenta MAS-ML e desenvolver com JAMDER 58
Figura 27 - É importante verificar a rastreabilidade dos artefatos gerados 59
Figura 28 - A planilha e o relatório de testes são úteis para apoiar os testes de SMAs com agentes racionais 59
Figura 29 - A escolha dos métodos/técnicas e ferramentas para apoio a modelagem de requisitos (iStar4RationalAgents), modelagem de projeto (MAS-ML 2.0) e framework de desenvolvimento (JAMDER) foi adequada 59
Figura 30 - O PSMAR pode ser útil para criar meus próximos projetos de sistemas multiagentes 60
Figura 31 - Qual o nível de dificuldade para entender o PSMAR 60
Figura 32 - Página do site https://jade.tilab.com/ 82
Figura 33 - Página de download do JADE 83
Figura 34 - Arquivos JAMDER, JADE e Eclipse 83
Figura 35 - Tabela de inicialização de projeto no Eclipse 84
Figura 36 - Selecionar propriedades no Eclipse 85
Figura 37 - Propriedades do Eclipse 86
Figura 38 - Extensões nas propriedades do Eclipse 86
Figura 39 - Extensões JADE e JAMDER 87
Figura 40 - Ferramenta Eclipse Modeling Tools 91
Figura 41 - Plugins MAS-ML 2.0 92
Figura 42 - Tela de criação de novo projeto no Eclipse 92
Figura 43 - Finalizando criação de novo projeto no Eclipse 93
Figura 44 - Inserindo classe no Eclipse 93

LISTA DE TABELAS

Tabela 1 - Comparação entre os processos relacionados com o trabalho proposto 21
Tabela 2 - Utilidade do PSMAR para a criação de sistemas multiagentes com agentes racionais 61
Tabela 3 - Adequação do PSMAR para futuros projetos de sistemas multiagentes com agentes racionais 61
Tabela 4 - Utilidade do PSMAR para níveis de experiência de desenvolvedores 61
Tabela 5 - Uso da proposta do PSMAR 61
Tabela 6 - Ambiente 75
Tabela 7 - Organização 76
Tabela 8 - Agente 76
Tabela 9 - Papéis de agentes 76
Tabela 10 - Relacionamento entre entidades fontes e entidades alvos 76
Tabela 11 - Entidade 1 76
Tabela 12 - Entidade 2 77
Tabela 13 - Entidade 3 77
Tabela 14 - Entidade 4 77
Tabela 15 - Planilha de mapeamento de construtores iStar4RationalAgents e MAS-ML 2.0 .. 78
Tabela 16 - Checklist de rastreabilidade - fases do processo concluídas 81
Tabela 17 - Checklist de rastreabilidade - entidades do processo concluídas 81
Tabela 18 - Planilha de teste [MODELO DE DOCUMENTO] 88
Tabela 19 - Planilha de teste [MODELO DE DOCUMENTO] - funcionalidade 01 88
Tabela 20 - Relatório de teste [MODELO DE DOCUMENTO] - Entidade 01 89
Tabela 21 - Relatório de teste [MODELO DE DOCUMENTO] - Entidade 02 89
Tabela 22 - Documento de gerenciamento de iterações [MODELO DE DOCUMENTO] 94

LISTA DE ABREVIATURAS E SIGLAS

BDI	Beliefs, Desires and Intentions
JAMDER	JAde to Multi agent systems DEvelopment Resource
JADE	Java Agent Development framework
MAS-ML	Multi-Agent System Modeling Language
MOODLE	Modular Object-Oriented Dynamic Learning Environment
PSMAR	Process to Support the development of Multi Agent systems with Rational agents
UAB	Universidade Aberta do Brasil
UML	Unified Modeling Language
SMA	Sistemas Multiagentes
TAO	Taming Agents and Objects

SUMÁRIO

1 INTRODUÇÃO 17

1.1 Objetivos 18

1.1.1 Objetivo Geral 18

2 TRABALHOS RELACIONADOS 19

3 FUNDAMENTAÇÃO TEÓRICA 22

3.1 Apoio ao Desenvolvimento de Agentes Racionais 24

3.1.1 iStar4RationalAgents 24

3.1.2 MAS-ML 2.0 30

3.1.3 JAMDER 33

4 PROCEDIMENTOS METODOLÓGICOS 36

4.1 Estudo das Técnicas Envolvidas 36

4.2 Definição da Especificação Textual de Requisitos 36

4.3 Definição do Mapeamento entre *iStar4RationalAgents* 36

4.4 Modelagem do PSMAR utilizando BPMN 37

4.5 Ilustração de uso do PSMAR 37

4.6 Avaliação do Processo Proposto 37

5 RESULTADOS 38

5.1 PSMAR 38

5.2 Fluxo Principal 38

5.3 Levantar e Especificar Requisitos do Sistema Multiagente 41

5.4 Gerar Documento de Gerenciamento de Iterações 42

5.5 Selecionar Requisitos a Serem Desenvolvidos 42

5.6 Detalhar Requisitos 42

5.7 Projetar Sistema Multiagente com MAS-ML 2.0 43

5.8 Desenvolver Sistema Multiagente com JAMDER 44

5.9 Testar Sistema Multiagente 46

5.10 Identificar Novos requisitos 47

6 ILUSTRAÇÃO DO USO DO PSMAR PARA DESENVOLVIMENTO DE SMA PARA O MOODLE 48

7 AVALIAÇÃO DO PSMAR 54

7.1 Universo e amostra dos participantes 54

7.2 Preparação para coleta 54

7.3 Coleta dos dados 55

7.4 Resultados da avaliação 55

7.5 Ameaças a validade 62

7.5.1 Validade de face 62

7.5.2 Validade de conteúdo 63

7.5.3 Validade de conclusão 63

7.5.4 Validade de construção 63

8 CONCLUSÃO 64

REFERÊNCIAS 66

APÊNDICE A – INSTRUMENTO DE COLETA DE DADOS 69

APÊNDICE B – DOCUMENTO DE REQUISITOS [MODELO DE DOCUMENTO] 75

APÊNDICE C – PLANILHA DE MAPEAMENTO DE CONSTRUTORES ISTAR4RATIONALAGENTS E MAS-ML 2.0 78

APÊNDICE D – CHECKLIST DE RASTREABILIDADE [MODELO DE DOCUMENTO] 81

APÊNDICE E – TUTORIAL DE INSTALAÇÃO DO JADE E JAMDER 82

APÊNDICE F – PLANILHA DE TESTE [MODELO DE DOCUMENTO] 88

APÊNDICE G – RELATÓRIO DE TESTE [MODELO DE DOCUMENTO] 89

APÊNDICE H – TUTORIAL DE INSTALAÇÃO DO PLUGIN MAS-ML TOOL 91

APÊNDICE I – DOCUMENTO DE GERENCIAMENTO DE ITERAÇÕES

[MODELO DE DOCUMENTO] 94

1 INTRODUÇÃO

Um agente é uma entidade de software capaz de perceber seu ambiente através de sensores e atuar nesse ambiente através de efetuadores (RUSSEL e NORVIG, 1995). Os agentes são sistemas de computador com duas importantes capacidades. Primeiro, eles são pelo menos até certo ponto capazes de ação autônoma – de decidir por si mesmos o que precisam fazer para satisfazer seus objetivos de projeto. Eles são capazes de interagir com outros agentes não simplesmente por troca de dados, mas envolvendo-se de forma análoga ao tipo de atividade social que nos utilizamos em nosso cotidiano: cooperação, coordenação e negociação (WOOLDRIDGE, 2002). Sistemas Multiagentes (SMA) é uma subárea de Inteligência Artificial (IA) que investiga o comportamento de um conjunto de agentes autônomos objetivando a solução de um problema que está além da capacidade de um único agente (JENNINGS, 1996).

Há diversas arquiteturas de agentes definidas na literatura, as quais representam o tipo/classificação de uma classe de agentes. RUSSEL e NORVIG (1995) apresentam quatro arquiteturas de agentes (agente reativo simples, agente reativo baseado em conhecimento, agente baseado em objetivo e agente baseado em utilidade) conhecidas como agentes racionais. O desenvolvimento de SMA com agentes racionais vem sendo tratado por soluções propostas a nível de modelagem de requisitos e projeto, e framework de desenvolvimento. *iStar4RationalAgents* (GONÇALVES et al. 2019) é uma extensão de *iStar 2.0* (DALPIAZ et al. 2016) para modelagem de SMA com agentes racionais no nível de requisitos. A linguagem de modelagem, *MAS-ML 2.0* (GONÇALVES et al. 2015) visa dar suporte à modelagem de SMA com agentes racionais a nível de projeto. *JAMDER* (LOPES et al. 2018) é um *framework* baseado em *JADE* (*Java Agent Development*) (*JADE*, 2020) para apoiar a codificação de SMA com agentes racionais.

Por outro lado, um processo de software é um conjunto de atividades que leva a produção de um produto de software. Essas atividades envolvem o desenvolvimento de software propriamente dito. As quatro atividades básicas do processo – especificação, desenvolvimento, validação e evolução – são organizadas de modos diferentes nos diversos processos de desenvolvimento de modo a atender suas particularidades (SOMMERVILLE, 2007).

Os trabalhos de GONÇALVES et al. (2019), GONÇALVES et al. (2015) e LOPES et al. (2018) são complementares, porém foram propostos de forma isolada, o que torna a integração das técnicas algo não trivial. iStar4RationalAgents apoia a fase de requisitos, por

meio da modelagem de conceitos gerais e enquanto o que vai ser desenvolvido está sendo definido. Uma vez que os requisitos estão definidos, MAS-ML 2.0 é utilizada para a especificação de um sistema multiagente a nível de projeto, apresentando detalhes de como a implementação deve ser realizada. Por fim, JAMDER dá apoio a implementação e conta com uma geração de código. Assim, estas são abordagens complementares e que tratam com um conjunto de conceitos de agentes racionais em diferentes etapas do processo de desenvolvimento.

Portanto, há a necessidade de sistematizar o uso destes resultados por meio da criação de uma abordagem que os integre e que preencha lacunas do desenvolvimento, como a especificação de requisitos de forma textual, o mapeamento entre a fase de requisitos e projeto e a definição de atividades de apoio a testes do SMA. Assim sendo, um processo de desenvolvimento pode contribuir neste contexto. Este trabalho propõe o PSMAR, um processo para conduzir o desenvolvimento de SMA com agentes racionais de modo a integrar as técnicas já existentes e propor tarefas necessárias ainda não abordadas por resultados anteriores

1.1 Objetivos

Nesta Seção, os objetivos deste trabalho serão apresentados.

1.1.1 Objetivo Geral

Apoiar o desenvolvimento de sistemas multiagentes com agentes racionais de forma integrada desde a especificação de requisitos até testes.

1.1.2 Objetivos Específicos

i. Integrar técnicas já existentes para desenvolvimento de sistemas multiagentes com agentes a nível de requisitos, projeto e implementação
ii. Propor tarefas/artefatos para as etapas do ciclo de desenvolvimento ainda não abordadas em trabalhos anteriores
iii. Modelar e disponibilizar o processo utilizando notação adequada
iv. Avaliar o processo criado com especialistas

2 TRABALHOS RELACIONADOS

Técnicas e ferramentas de engenharia de software são necessárias ao longo do ciclo de vida do software, e se fazem relevantes para conduzir com êxito o processo de desenvolvimento de softwares. A engenharia de software orientada a agentes tem como objetivo prover técnicas e ferramentas para o desenvolvimento de SMA (CASTRO *et al.* 2006). Os trabalhos a seguir são abordagens que apoiam o desenvolvimento de SMA.

MAS-CommonKADS (IGLESIAS, C. A.; GARIJO, 2005) é uma metodologia da engenharia do conhecimento amplamente utilizada que estende *CommonKADS* (SCHREIBER *et al.* 2000). Segundo (SCHREIBER *et al.* 2000), *MAS-CommonKADS* é composta de cinco fases, que vão desde a contextualização do sistema até o desenvolvimento e manutenção. As fases do processo são: Contextualização, Análise, Projeto, Desenvolvimento e Teste, Operação e Manutenção. *MAS-CommonKADS* possui sete modelos que buscam descrever as características de um agente e seus comportamentos sociais no SMA: modelo de agente, modelo de tarefas, modelo de coordenação, modelo de comunicação, modelo de experiência, modelo de organização e modelo de projeto. *MAS-CommonKADS+* (MORAIS II, 2010) é uma extensão que acrescenta a representação de agentes racionais a nível de projeto ao *MAS- CommonKADS*.

MESSAGE (HENDERSON-SELLERS e GIORGINI, 2005) segue a abordagem iterativa e incremental do *RUP*, que envolve aprimoramento progressivo dos requisitos, plano, design e implementação. Em seu processo de desenvolvimento o *MESSAGE* consiste nas fases de análise e projeto. O objetivo da fase de análise é gerar uma especificação do sistema que descreve o problema a ser solucionado, de modo a gerar um modelo abstrato que auxilia na validação e no desenvolvimento do projeto (HENDERSON-SELLERS e GIORGINI, 2005). Essa fase gera cinco modelos: modelo de organização, modelo de agentes/papéis, modelo de metas/tarefas, modelo de domínio, modelo de interação. A fase de projeto transforma os modelos gerados na análise em entidades que podem ser codificadas em uma plataforma de agentes. Esta fase é dividida em projeto de alto nível e projeto detalhado. O projeto de alto nível especifica os agentes do sistema e quais papéis são de sua responsabilidade e o projeto detalhado busca mapear os modelos de alto nível para conceitos computacionais específicos de plataformas.

Prometheus (PADGHAM; WINIKOFF, 2005), é uma metodologia de desenvolvimento que fornece mecanismos para a análise e projeto de SMA baseados em arquiteturas BDI. O processo de desenvolvimento é dividido em três fases: especificação do sistema, projeto arquitetural e projeto detalhado. A fase de especificação do sistema tem como

finalidade definir os requisitos do sistema por meio da descrição das metas do sistema, dos cenários, das funcionalidades e do ambiente, que é descrito através de suas percepções, ações e dados externos. O projeto arquitetural é onde os artefatos gerados na fase anterior são modelados por meio de diagramas de acoplamento e de conhecimento, diagramas de interação e do diagrama geral do sistema. A fase de projeto detalhado tem como objetivo descrever o projeto interno do agente através de suas capacidades, planos eventos e dados. Uma capacidade pode conter planos, eventos e dados.

A tabela 1 faz uma comparação entre os processos das metodologias abordadas nos trabalhos relacionados com o trabalho proposto. Na primeira linha (cabeçalho) são listadas as abordagens: *MAS-CommonKADS+*, *MESSAGE*, *Prometheus*, PSMAR.

Na segunda linha, é descrito se a respectiva abordagem foi modelada utilizando linguagem de modelagem de processo, sendo que apenas o PSMAR cumpre com este item. Na terceira linha, é verificada a cobertura da tarefa de especificação textual de requisitos, sendo que somente o PSMAR cumpre com este item da análise.

Na quarta linha, é analisado se cada abordagem define a modelagem de requisitos, sendo identificado que todas as abordagens contemplam. Na quinta linha, é verificado que a modelagem de projeto é tratada em todas as abordagens analisadas. Na sexta linha, é destacado que a geração de código é utilizada somente pelo PSMAR dentre as abordagens analisadas.

Na sétima linha da Tabela 1 é analisada a presença de fase de testes do SMA no qual apenas a metodologia *MESSAGE* e PSMAR abrangem esta fase. Por fim, a oitava e última linha analisa se cada uma das abordagens trata de agentes racionais, sendo que apenas *MAS-CommonKADS+* e PSMAR forma assinaladas. Vale destacar que, *MAS-CommonKADS+* aplicou os conceitos de agentes racionais somente a nível de modelagem de requisitos e de projeto.

Tabela 1 - Comparação entre os processos relacionados com o trabalho proposto

Análise/Abordagem	*MAS-CommonKADS+*	*MESSAGE*	*PROMETHEUS*	PSMAR
Modelada com linguagem de modelagem de processo	✗	✗	✗	✓
Especificação textual de requisitos	✓	✗	✓	✓
Modelagem de projeto	✓	✓	✓	✓
Geração de código	✓	✓	✓	✓
Atividade de testes	✗	✓	✗	✓
Agentes racionais	✓	✗	✗	✓

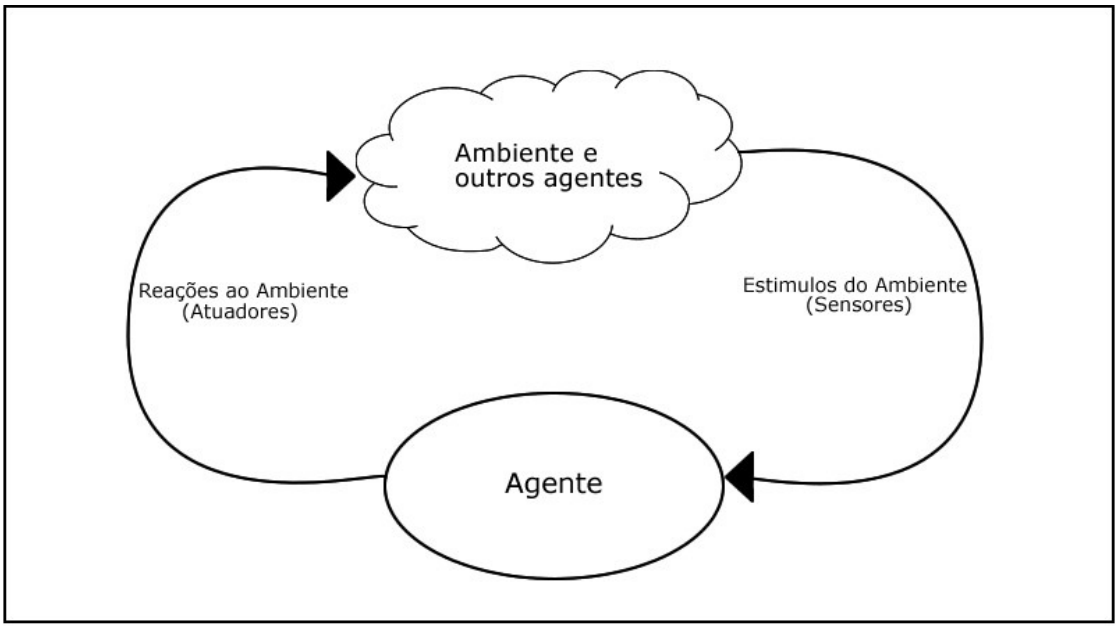
Fonte: Elaborada pelo autor.

3 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo serão apresentados os conceitos utilizados para a realização deste trabalho. A Seção 3.1 descreve conceitos relacionados a SMA e agentes racionais, e a Seção 3.2 apresenta trabalhos que apoiam o desenvolvimento de SMA com agentes racionais.

Um agente é um sistema computacional situado em um ambiente e que é capaz de efetuar ações autônomas neste ambiente a fim de alcançar seus objetivos (WOOLDRIDGE, 2007). Agente racional é definido como uma entidade de software autônoma que percebe seu ambiente de atuação por meio de sensores, processa essas informações e conhecimentos, e atua no ambiente por meio de atuadores visando realizar algum objetivo, conforme estabelecido em uma medida de avaliação de desempenho especificada pelo projetista do agente (RUSSELL e NORVIG, 1995). A Figura 1 ilustra esta definição.

Figura 1 - Visão geral de um agente



Fonte: MORAIS II, 2010.

Existem agentes com outros tipos arquitetura interna que podem ser entendidos como racionais (como BDI, por exemplo), porém no escopo deste trabalho consideramos as arquiteturas de agentes racionais definidas por Russell e Norvig por conta dos trabalhos em que este trabalho se baseia terem sido propostos baseados nessas arquiteturas.

Franklin e Graesse (1996), definem algumas propriedades a serem observadas em um agente computacional, são elas (WOOLDRIDGE, 2002):

- **Autonomia –** os agentes devem executar a maior parte de suas ações sem interferência direta de agentes humanos ou de outros agentes computacionais, possuindo controle total sobre suas ações e estado interno.
- **Habilidade Social –** os agentes devem poder interagir com outros agentes (humanos ou computacionais), para completarem a resolução de seus problemas, ou ainda para auxiliarem outros agentes.
- **Reatividade -** os agentes devem perceber e reagir a alterações nos ambientes em que estiverem inseridos.
- **Proatividade –** os agentes devem atuarem em resposta às alterações ocorridas em seu ambiente, devem apresentar um comportamento orientado a objetivos, tomando iniciativas quando consideram apropriado.
- **Adaptação -** os agentes devem poder mudar o seu comportamento devido a uma experiência anterior.
- **Mobilidade –** os agentes devem poder se movimentar de uma máquina para outra.
- **Persistência ou continuidade temporal –** capacidade do agente de manter um estado interno conciso através do tempo, isto é, o agente estar continuamente executando um processo.

Alguns dos motivos para utilização de sistemas multiagentes são (BITTENCOURT, 1998):

- Aumentar a eficiência e a velocidade do sistema.
- Melhorar a adaptabilidade, a confiabilidade e a autonomia do sistema.
- Permitir a integração de sistemas inteligentes existentes de maneira a aumentar a capacidade de processamento e, principalmente, a eficiência na solução de problemas.
- Reduzir os custos de desenvolvimento e manutenção.

Russel e Norvig (1995) descrevem quatro arquiteturas de agentes que levam em consideração não apenas como realizar suas ações, mas também as informações de como o agente irá evoluir e como aquelas ações irão modificar o ambiente, definindo assim um conceito mais racional do agente, são elas:

- Agentes reativos simples: são os agentes mais simples que percebem o ambiente e agem baseados nas suas percepções atuais sem considerar o histórico de suas percepções. Geralmente são providos com uma base de conhecimento formada

por regras Se-Então, sendo que seu comportamento está totalmente codificado nessas regras.

- Agentes reativos baseados em modelos: são agentes que guardam estados do ambiente e que sabem como o ambiente evolui em função do tempo e em função de suas ações. Portanto as regras de produção podem se basear tanto na sequência de percepção quanto no estado do ambiente para decidir o que agente deve fazer.
- Agentes baseados em objetivos: este tipo de agente toma suas decisões sempre levando em consideração a tentativa de alcançar seus objetivos.
- Agentes baseados em utilidade: pode acontecer de existirem várias sequências distintas de ações que levem o agente a atingir seu objetivo. Então esse tipo de agente toma suas decisões baseado na função de utilidade que melhor se adeque para a resolução do seu objetivo.
-

3.1 Apoio ao Desenvolvimento de Agentes Racionais

Nesta seção serão apresentados alguns trabalhos que apoiam o desenvolvimento de SMA com agentes racionais. Na subseção 3.2.1 é detalhada a extensão *iStar4RationalAgents*. Na subseção 3.2.2 *MAS-ML 2.0* é apresentada. *JAMDER*, um framework baseado em *JADE* para desenvolvimento de SMA com agentes racionais, é apresentado na subseção 3.2.3.

3.1.1 iStar4RationalAgents

O *iStar4RationalAgents* é uma extensão proposta à linguagem de modelagem *iStar*. O *iStar* é uma linguagem de modelagem baseada em objetivos, usada para modelar software no nível de requisitos (GONÇALVES et al. 2019).

Na estrutura do *iStar*, as partes interessadas são representadas como atores que dependem uma da outra para atingir seus objetivos, executar tarefas e fornecer recursos. Cada objetivo é analisado do ponto de vista do ator, resultando em um conjunto de dependências entre pares de atores. Os elementos são classificados como elementos intencionais (objetivo, tarefa e recurso), atores (ator geral, função, cargo e agente) e links (links de meios finais, decomposição, contribuição e ator). Esses elementos são representados em dois modelos: Dependência estratégica (*SD*) e Raciocínio estratégico (*SR*). O modelo *SD* descreve os links e dependências externas entre os atores organizacionais. O modelo *SR* permite uma análise de

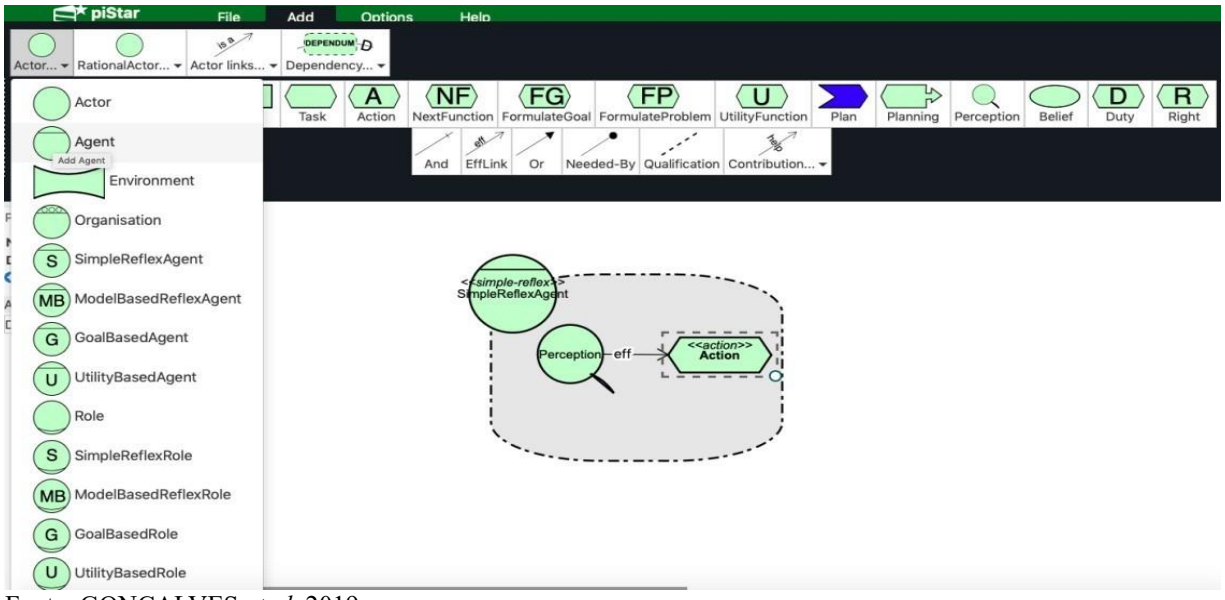
como os objetivos podem ser alcançados através de contribuições dos diversos atores (GONÇALVES et al. 2019).

O modelo de *SD* deve ser criado para representar os agentes, papéis, organizações e ambientes envolvidos no SMA e representar o relacionamento entre eles. Um agente pode desempenhar um papel, habitar um ambiente e fazer parte de uma organização. As dependências entre agentes, funções, organizações e ambientes também podem ser expressas. O principal objetivo desta modelagem é representar o SMA a ser desenvolvido na fase inicial, quando o SMA começar a ser proposto e as decisões sobre os elementos intencionais dos agentes, como objetivo, tarefa, planejamento, percepção e detalhes internos das intenções dos agentes elementos ainda não estão claros (GONÇALVES et al. 2019).

Além disso, o modelo *SR* deve ser criado para representar os detalhes internos dos agentes, papéis, organizações e ambientes envolvidos no SMA e representar o relacionamento entre eles. O modelo *SD* é o ponto de partida para criação deste modelo (GONÇALVES et al. 2019).

A extensão foi aplicada na ferramenta de modelagem *piStar*. A ferramenta estendida foi testada pela modelagem da ilustração, não foram encontradas correções a serem feitas. A ferramenta está disponível em www.cin.ufpe.br/~ler/piStar4rationalagents. A Figura 2 mostra uma visão geral do *piStar4RationalAgents* (GONÇALVES et al. 2019).

Figura 2 - Visão geral da ferramenta piStar4RationalAgents



Fonte: GONÇALVES *et al.* 2019

Na ilustração foi modelado o SMA para o MOODLE com a extensão proposta. O MOODLE (Modular Object-Oriented Dynamic Learning Environment - Ambiente modular de

aprendizagem dinâmica orientada a objetos) é amplamente utilizado em cursos à distância. Este software está disponível em uma distribuição padrão que pode ser personalizada. Esse ambiente é utilizado em cursos da Universidade Aberta do Brasil (UAB), uma universidade do governo brasileiro que oferece cursos em educação a distância. Assim, foi modelado um SMA para os cursos da UAB oferecidos em parceria com a Universidade Estadual do Ceará (UAB/UECE) (GONÇALVES et al. 2019).

Cinco tipos de agentes foram consideradas na modelagem:

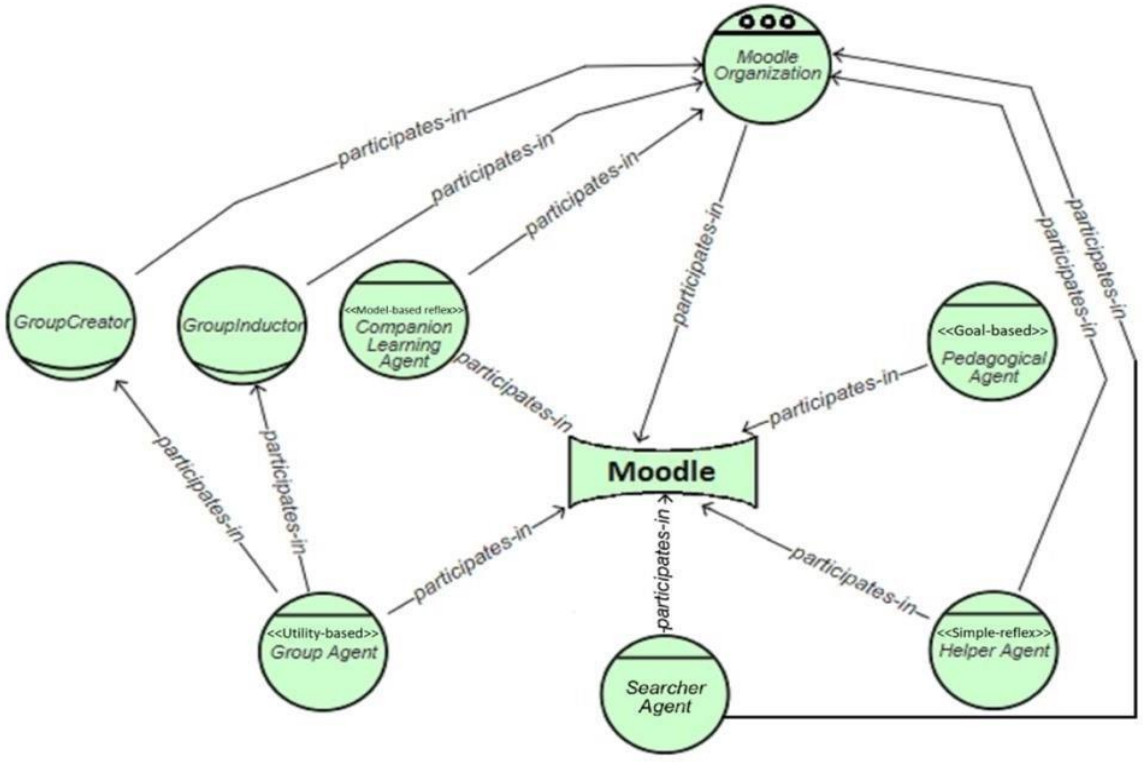
- **Agente Auxiliar:** Este agente é um tipo simples de reflexivo e possui uma lista de percepções sobre as dificuldades que o usuário possui e, antes disso, escolhe a ação apropriada. Esse agente percebe em que momento o usuário está e, ao mesmo tempo, oferece dicas sobre como fazer o melhor uso de uma funcionalidade específica, especificamente, a ação para um trabalho específico.
- **Agente de Aprendizado Associado:** Esse tipo de agente deve poder escolher independentemente entre um intervalo predeterminado de estratégias de interação afetiva, como mensagens de suporte. Apresenta mensagens encorajadoras (reforço positivo) quando o usuário, por meio das interações manifestas, fornece evidências fáceis de seguir a discussão e / ou as tarefas e / ou conteúdos propostos, e mesmo quando o aluno apresenta notas e iterações acima da média em sua classe ou grupo de trabalho. Devido à necessidade de manter anotações de aula para comparação e enviar mensagens rapidamente, esse agente é caracterizado como um agente reflexivo baseado em modelo.
- **Agente Pedagógico:** Esse agente deve poder acompanhar o aluno nas diferentes disciplinas que participam para contribuir com o usuário através de dicas, sugestões e mensagens relacionadas ao tópico em andamento e não apenas à natureza afetiva das mensagens (suporte). É um agente baseado em objetivos, porque precisa criar uma estratégia de estudo, sugerindo disciplinas para o aluno com base nas disciplinas que o aluno está realizando. Este agente é modelado como um agente baseado em objetivos.
- **Agente do grupo:** Este agente é um agente baseado em utilidade. Deverá ser capaz de ajudar autonomamente usuários, estudantes e educadores, na composição de grupos de trabalho, levando em consideração temas de afinidade ou perfis de aprendizagem. Para isso, deve considerar certos critérios

estabelecidos por um treinador de uma ou mais classes ou pelo usuário interessado em integrar os grupos de trabalho.

- **Agente de Pesquisa:** Este agente encontra material extra (páginas, projetos e outros objetos digitais) relacionados aos cursos no MOODLE e enviado aos alunos. Este agente é modelado como um agente padrão do *MAS-ML*.

Inicialmente, o engenheiro de requisitos deve modelar o diagrama estendido para *SD* com a representação do tipo de agentes com o estereótipo específico. Seus papéis devem ser representados e também a organização e o ambiente. Por fim, o link participates-in deve ser usado para representar os papéis desempenhados pelos agentes e o ambiente em que eles habitam. A participação também é usada para representar a propriedade dos agentes por suas organizações. O diagrama *SD* do SMA do MOODLE representa o MOODLE como o ambiente em que os agentes podem perceber e agir. Foi modelada uma organização para coordenar os comportamentos dos agentes. Finalmente, o diagrama representa quatro agentes e duas funções desempenhadas por dois deles. A Figura 3 mostra o diagrama *SD* para o SMA do MOODLE (GONÇALVES et al. 2019).

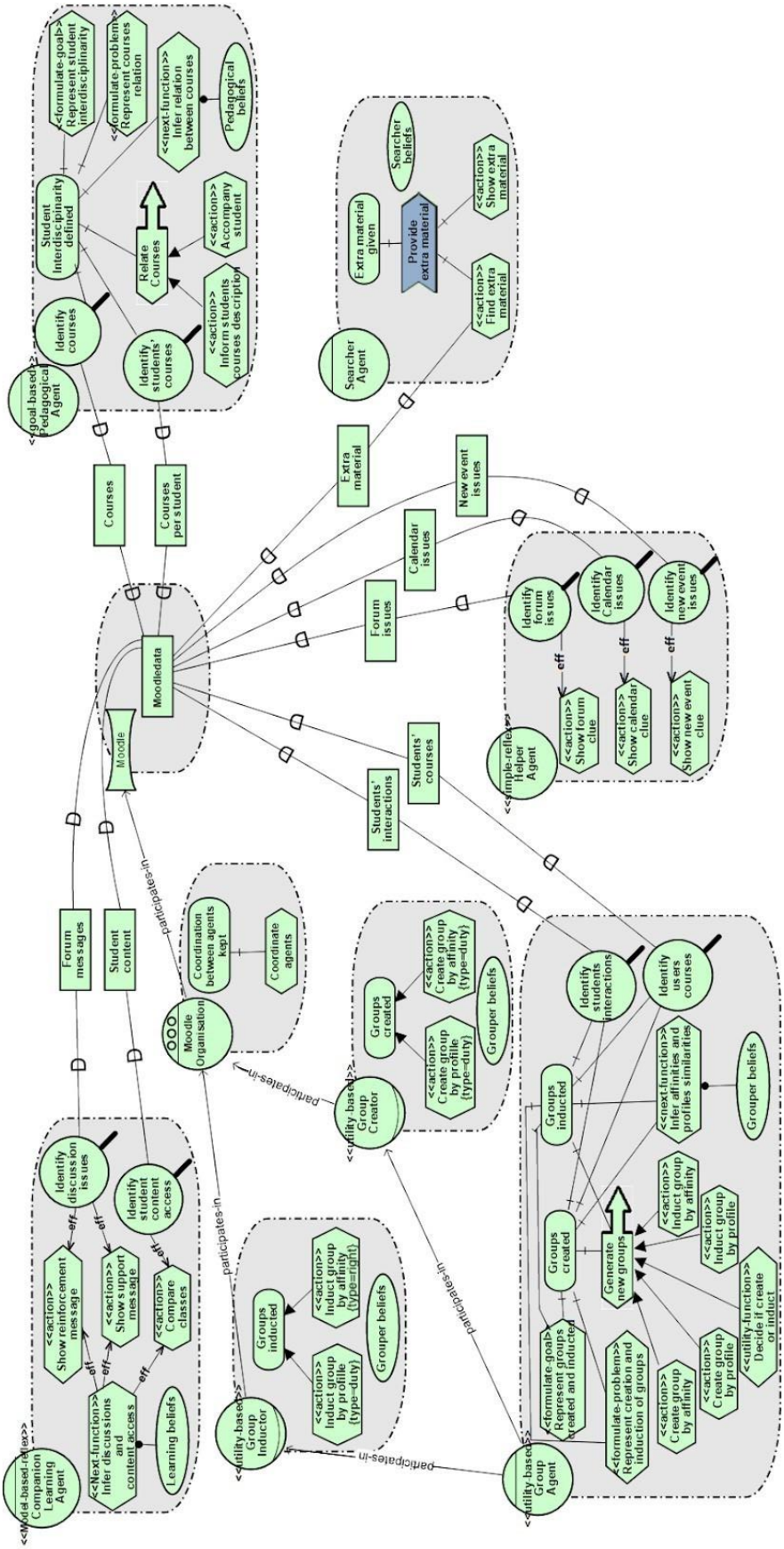
Figura 3 - Modelagem do modelo SD do iStar para o MOODLE usando a nova extensão (GONÇALVES, *et al.* 2019)



Fonte: GONÇALVES *et al.* 2019.

O diagrama *SR* mostra os elementos internos dos agentes. O conjunto de elementos relacionados depende do tipo de agente. As percepções dos agentes estão relacionadas ao meio ambiente por um relacionamento de dependência. A Figura 4 mostra o diagrama *SR* para o SMA do MOODLE (GONÇALVES et al. 2019).

Figura 4 - Modelagem do modelo SR do iStar para o MOODLE usando a nova extensão (GONÇALVES *et al*. 2019)



Fonte: GONÇALVES *et al.* 2019.

3.1.2 MAS-ML 2.0

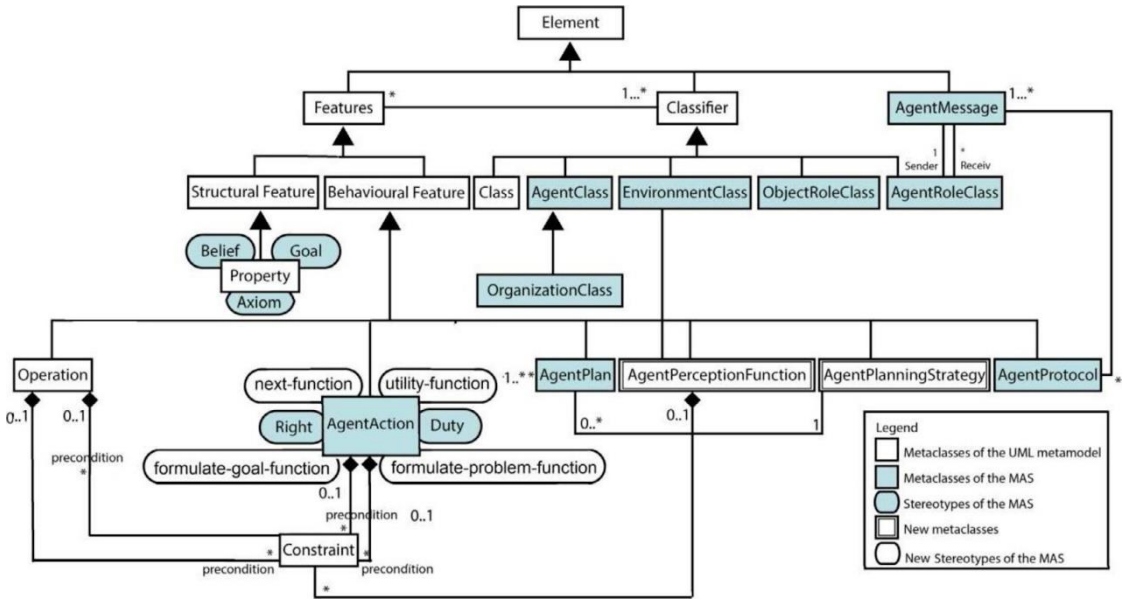
MAS-ML 2.0 é uma linguagem de modelagem que implementa os conceitos de *TAO* (*Taming Agents and Objects*) e foi originalmente projetada para suportar a modelagem de agentes proativos baseados em objetivos, guiados por planos pré-estabelecidos (SILVA, 2004). *MAS-ML* modela todos os aspectos estruturais e dinâmicos definidos no metamodelo *TAO* (*Taming Agents and Objects*) estendendo o metamodelo *UML* (*Unified Modeling Language*). *MAS-ML* modela cinco diagramas, os quais são extensões dos diagramas de *UML*. Os diagramas estruturais definidos pelo *MAS-ML* são: Diagrama de Classes, Diagrama de Organização e Diagrama de Papéis. Os diagramas dinâmicos são Diagrama de Sequência e Diagrama de Atividades. Em *MAS-ML* agentes são modelados com objetivos, crenças, planos e ações.

MAS-ML 2.0 (GONÇALVES, 2009) é uma extensão de *MAS-ML* para incluir elementos internos de agentes racionais. A extensão é utilizada para apoiar a modelagem de agentes usando diferentes arquiteturas internas: Reflexivo Simples, Reflexivo Baseado em Modelo, Baseado em Objetivo e Baseado em Utilidade.

Segundo MILES e HAMILTON (2006), valores marcados, estereótipos e restrições são mecanismos de extensão. Além disso, a adaptação das metaclasses existentes e a definição de novas metaclasses também podem ser usadas. Os estereótipos e a definição de novas metaclasses foram usados para representar agentes reflexos simples, agentes reflexivos baseados em modelos, agentes baseados em objetivos com agentes planejados e baseados em serviços públicos. Seguindo as definições de arquitetura apresentadas na pesquisa, sentiram a necessidade de definir as seguintes características para a extensão: percepção, função seguinte, função formular meta, função formular problema, função formular problema, planejamento e função utilidade (GONÇALVES et al., 2015).

A Figura 5 ilustra o metamodelo *MAS-ML 2.0*. As novas metaclasses são representadas por meio usando retângulos brancos de linha dupla, já os novos estereótipos são representados com um retângulo branco com bordas arredondadas (GONÇALVES et al. 2015).

Figura 5 - Metamodelo da extensão do MAS-ML 2.0



Fonte: GONÇALVES *et al.* 2015.

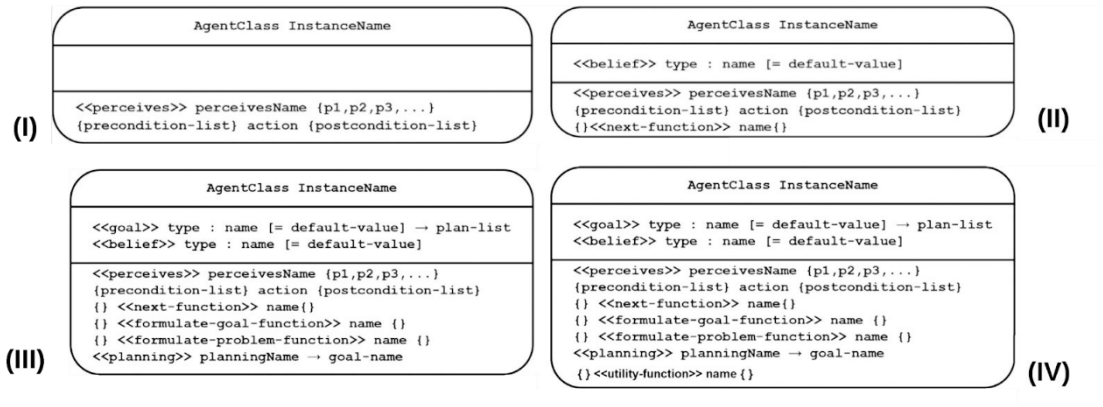
Os novos recursos estruturais e comportamentais introduzidos no metamodelo *MAS-ML* são usados para modelar os diferentes tipos de agentes e impactam a representação da metaclasse *AgentClass* em diagramas estáticos (GONÇALVES et al. 2015). Semelhante aos diagramas estáticos, as novas definições de agente também impactam os diagramas de sequência e atividade.

Na Figura 6 são representadas as modelagens dos agentes nos diagramas estáticos. Na Figura 6-I o Agente Reativo Simples não incluiu nenhum elemento estrutural no compartimento intermediário, mas no compartimento inferior são representadas as percepções e ações, orientadas por regras de condição ação e não por um plano específico. Na Figura 6-II o Agente Reativo Baseado em Conhecimento é representado. Assim, há a representação de crenças representando o estado, além de percepções, ações e da função próximo. Na Figura 6-III o Agente Baseado em Objetivo é apresentado. O qual possui objetivos e crenças no compartimento intermediário, e percepções, ações, função próximo, função de formulação de objetivo, função de formulação de problema e planejamento no compartimento inferior.

Na Figura 6-IV é representado o Agente Baseado em Utilidade, que consiste em uma especialização do Agente Baseado em Objetivo com planejamento. Durante o planejamento, os agentes podem ser vinculados para atingir mais de uma meta. Nesse caso, é possível a ocorrência de objetivos conflitantes ou a existência de vários estados que atendem aos objetivos. Portanto, a função de utilidade é incorporada à estrutura do agente para avaliar o

grau de utilidade dos objetivos associados. Assim. O elemento <<*utility-function*>> é adicionado para representar a função utilidade responsável pela otimização do desempenho do agente.

Figura 6 - Representação dos agentes com diagrama estáticos

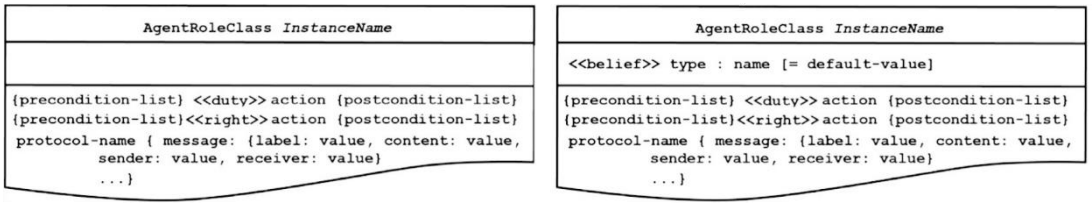


Fonte: GONÇALVES *et al.* 2015.

Um papel de agente é representado em *MAS-ML* por um retângulo sólido com uma curva na parte inferior e possui três compartimentos separados por linhas horizontais. O compartimento superior contém o nome do papel do agente, o compartimento intermediário contém uma lista de objetivos e crenças associadas ao papel e, abaixo, uma lista de deveres, direitos e protocolos (SILVA, 2004).

Os agentes reativos não têm objetivos explícitos e, mais particularmente, os agentes reflexivos simples não têm crenças. Assim, sua representação de papéis foi adaptada por GONÇALVES et al. (2015) e é ilustrada na Figura 7, sendo o papel de agente reativo simples do lado esquerdo e de agente reativo baseado em conhecimento do lado direito.

Figura 7 - Representação de papéis para agente reflexivo simples e agente reativo baseado em conhecimento



Fonte: GONÇALVES *et al.* 2015.

A ferramenta *MAS-ML tool* (Silva *et al*. 2007); (GONÇALVES *et al.* 2015) é um *plug-in* do Eclipse para apoiar a modelagem da linguagem *MAS-ML 2.0* (GONÇALVES et al.

2011). *MAS-ML tool* suporta a modelagem dos seguintes diagramas: diagrama de organização, diagrama de funções e diagrama de classes, com base no metamodelo *MAS-ML*. Essa ferramenta gera o arquivo .masml (*XMI - XML Metadata Interchange*) que armazena a estrutura das entidades de dados e os aspectos estruturais e comportamentais definidos no *MAS-ML 2.0.*

3.1.3 JAMDER

JAMDER é uma extensão do *JADE*, um *framework* gratuito que possui uma plataforma na linguagem de programação JAVA e que também suporta sistemas distribuídos. No entanto, a implementação do agente no *JADE* é limitada principalmente a agentes, comportamentos e mensagens. O *JADE* também possui uma plataforma, que contém o ambiente necessário para o ciclo de vida do agente e contêineres em que os agentes residem. Esta extensão fornece a infraestrutura adequada focada na implementação de agentes de acordo com as configurações das arquiteturas internas de agentes racionais (LOPES et al, 2018).

Além disso, contém uma padronização da geração de código. O principal benefício da extensão proposta é incluir as arquiteturas, entidades e relacionamentos internos do agente em uma estrutura de implementação e aumentar a produtividade por geração de código, garantindo a consistência entre o design e o código (LOPES et al, 2018).

A estrutura *JADE* inclui classes e serviços para facilitar a fase de codificação do *MAS*. Dentre os recursos disponíveis, alguns estão listados a seguir: serviços de publicação em páginas amarelas, serviço de localização em páginas brancas, suporte a ontologias e comunicação de protocolos compatíveis com o padrão *FIPA 6* (*Foundations of Intelligent Physical Agents*). De fato, o *JADE* é uma estrutura orientada a objetos escrita em Java (LOPES et al, 2018).

A arquitetura no *JADE* contém contêineres onde os agentes residem e o sistema pode ser distribuído em diferentes plataformas. Cada agente é registrado no serviço *AMS* (*Agent Management System*) fornecido pela *JADE* que garante a unicidade dos agentes. Para encontrar outros agentes, outro serviço é fornecido, o *DF* (*Directory Facilitator*) e funciona como um serviço de Páginas Amarelas (LOPES et al, 2018).

O ciclo de vida do *JADE* para um agente é definido de acordo com o *FIPA*. A primeira etapa é a execução do construtor do agente, seguida pela atribuição de um identificador para o agente a ser inserido no sistema. O método *setup()* é executado a partir do momento em que o agente inicia suas atividades. Um comportamento do agente é projetado pela substituição da configuração (LOPES et al, 2018).

A arquitetura orientada a modelo (MDA) é apresentada como uma abordagem apropriada para auxiliar na geração de código, porque o código pode ser gerado várias vezes sem comprometer o modelo. O *OMG* 7 define alguma padronização nesse processo que não está necessariamente associada a uma plataforma específica. Assim, os conceitos podem ser aplicados a diferentes linguagens de modelagem e implementação (LOPES et al, 2018).

O processo de transformação ocorre através das etapas propostas pela *OMG*. Os artefatos gerados em cada uma dessas etapas são independentes de uma ferramenta específica a ser usada, por exemplo, quando o mesmo aplicativo pode ser implementado em linguagens diferentes. Os conceitos de cada uma dessas etapas são descritos a seguir (LOPES et al, 2018):

- *CIM* (*Computation Independent Model*): regras para requisitos funcionais;
- *PIM (Platform-Independent Model*): relações entre propriedades e seus relacionamentos com entidades.
- *PSM* (*Platform-Specific Model*): definição de como o sistema funcionará em uma plataforma específica.
- *PDM* (*Platform Description Model*): definição de como o *PIM* para *PSM* funcionará.

No contexto da abordagem MDA é proposto o suporte automatizado à geração de código para o SMA. No processo proposto, são aplicados os seguintes componentes para gerar código: Ferramenta *MAS-ML tool* (*PIM*); *Framework* Java / *JAMDER* (*PSM*); e modelos *Acceleo* (*PDM*). Os arquivos de modelagem gerados pela ferramenta *MAS-ML tool* são a entrada para o processo de geração de código usando o *plug-in* Acceleo para suportar o conceito de *MDA*, pois permite a geração de código em diferentes linguagens de código e o desenvolvimento incremental (LOPES et al, 2018).

Para formalizar a geração de código no *Acceleo,* é necessário estabelecer um modelo para cada entidade por meio de uma linguagem definida pelo OMG, o *MTL* (*Model Transformation Language*). Quando o modelo é executado no Eclipse, ele precisará da representação do modelo *MAS-ML tool* (arquivos *.masml*) e da pasta de saída para armazenar as classes *JAMDER*) (LOPES et al, 2018).

Por exemplo, o ambiente é uma instância de uma classe que herda da classe *Environment* no *JAMDER* e sua representação ocorre através do *EnvironmentClass* na ferramenta *MAS-ML tool*. Como essa instância herda os métodos definidos em Ambiente, é necessário apenas, na geração do código, o construtor dessa classe chamar a superclasse e criar organizações, agentes e funções de agente nessa ordem. A necessidade de criar instâncias de função de agente é apenas para vincular a organização e as instâncias do agente, nas quais a construção da função de agente faz esse link. O mapeamento de agentes de criação e

organização de corpos ocorre por meio da relação entre o ambiente e essas entidades. Os objetos e funções de objeto também são criados no ambiente em que as funções são obtidas pela organização por meio do relacionamento de propriedade. Por sua vez, os objetos são alcançados pelo objeto através do relacionamento lúdico. A Figura 8 mostra o código gerado para um ambiente modelado em MAS-ML tool.

Figura 8 - Environment template (Lopes, *et al.* 2018)

```
public class (c.name /) extends Environment {
  public (c.name /) (String name, String host, String port) {
        super(name, host, port);
   (for(i : Inhabit | c.inhabit))
       (let organ : OrganizationClass = i.org.name)
        (if (i.org -> size() > 0) )
          Organization (organ.name/) = new Organization("(organ.name/)", this, null);
          addOrganization("(organ.name/)", (organ.name/));
          (if (i.org.play -> size() > 0) )
          (let ar : AgentRoleClass = i.org.play.agentRole)
            AgentRole (ar.name/) = new AgentRole("(ar.name/)",
(ar.ownership.owner.name/), (organ.name/));
          (/let)  (/if)

          (for(ow : Ownership | i.org.ownership))
                   (for(ob : ObjectRoleClass | ow.objectRole))
                     Object (ob.play.name/) = new Object();
                     addObject("(ob.name/)", (ob.name/));
                     ObjectRole (ob.name/) = new ObjectRole("(ob.name/)",
(organ.name/), (ob.play.name/));
                   (/for)
          (/for)
           (/if)
      (/let)
    (/for)

    (for(i : Inhabit | c.inhabit))
       (if (i.agentClass -> size() > 0) )
          (let a : AgentClass = i.agentClass)
         GenericAgent (a.name /) = new (a.name /)("(a.name /)", this, null);
          AgentRole (a.play.agentRole.name/) = new
AgentRole("(a.play.agentRole.name/)", (a.play.agentRole.ownership.owner.name/),
(a.name/));
          addAgent("(a.name /)", (a.name /));
          (/let)
       (/if)
    (/for)
  }
  // Additional attributes
  (for(p : Property | c.ownedProperty))
     (p.visibility/) (p.type.toString()/) (p.name/);
  (/for)

  // Additional methods
  (for(o : Operation | c.ownedOperation))
     (o.visibility /) (o.returnValue /) (o.name /)
        ((for(p:Parameter | o.parameter) separator(',')) (p.type/) (p.name/) (/for)) {}
  (/for)
}
(/file)
(/template)
```

Fonte: LOPES *et al.* 2018.

4 PROCEDIMENTOS METODOLÓGICOS

Neste capítulo serão descritos os passos para a execução dos artefatos referentes a este trabalho.

I. Estudo das técnicas envolvidas.
II. Definição da especificação textual de requisitos.
III. Definição do mapeamento entre *iStar4RationalAgents e MAS-ML 2.0*.
IV. Modelagem do PSMAR utilizando *BPMN*.
V. Ilustração de uso do PSMAR.
VI. Avaliação do processo proposto.

4.1 Estudo das Técnicas Envolvidas

Nesta etapa foi realizado o estudo de modelos de processo de software, onde cada modelo de processo representa um processo sob determinada perspectiva, e dessa forma, viabilizar a criação do modelo proposto fundamentando nas variações desses processos e então estabelecer seus principais estágios que definem as atividades fundamentais de desenvolvimento. Além disto, os trabalhos existentes de *iStar4RationalAgents*, *MAS-ML 2.0* e *JAMDER* foram estudados.

Um SMA para o MOODLE foi desenvolvido para o MOODLE utilizando estas técnicas (GONÇALVES, ARAUJO e CASTRO, 2019) e (GONÇALVES *et al.* 2014). Os passos seguidos neste desenvolvimento foram analisados de modo a servir como base para a modelagem do processo PSMAR. Foi identificado, além do uso das técnicas já mencionadas, o uso de uma planilha de testes utilizada durante o desenvolvimento do SMA. Vale destacar a ausência de uma especificação textual de requisitos e de um mapeamento entre *iStar4RationalAgents* e *MAS-ML 2.0*.

4.2 Definição da Especificação Textual de Requisitos

Nesta etapa foram estabelecidas as atividades para sistematizar a especificação dos requisitos do SMA com agentes racionais. Foi criado uma abordagem de especificação textual baseada em diferentes níveis de especificação.

4.3 Definição do Mapeamento entre *iStar4RationalAgents*

A modelagem do SMA com agentes racionais envolve a modelagem de requisitos com *iStar4RationalAgents* e *MAS-ML 2.0*. É importante portanto mostrar um mapeamento entre as entidades destas linguagens para facilitar a conversão entre os modelos. O mapeamento entre modelos de *iStar* e de *UML* apresentado por MELO *et al.* (2015) foi utilizado como ponto de partida.

4.4 Modelagem do PSMAR utilizando BPMN

Nesta etapa foi feita a criação do processo de desenvolvimento de SMA com agentes racionais, utilizando *BPMN*, uma notação de metodologia de gerenciamento de processos de negócio. Foi adotada a notação *BPMN* considerando que a mesma permite que seja ilustrado e compreendida todas as tarefas operacionais de um negócio de forma lógica, sequencial, e também é possível identificar os papéis de cada um, os eventos e todos os demais componentes de um processo.

Fazendo-se possível que profissionais de diversas nacionalidades consigam compreender a dinâmica de um fluxo operacional por meio dos símbolos utilizados pela notação.

4.5 Ilustração de uso do PSMAR

A ilustração do PSMAR consiste no uso do processo proposto para desenvolver um SMA. Assim, um SMA para acompanhamento de cursos no MOODLE foi proposto.

4.6 Avaliação do Processo Proposto

A avaliação do PSMAR foi realizada com especialistas em SMA por meio de um questionário online auto aplicado. As diretrizes estabelecidas por KITCHENHAM e PFLEEGER (2002) foram seguidas neste estudo.

5 RESULTADOS

5.1 PSMAR

Esse processo é o foco elementar desta pesquisa; conduz metodicamente a metodologia de desenvolvimento, garantindo a integridade, consistência e, com intuito de reduzir falhas durante a criação de SMA com agentes racionais. A versão completa do processo pode ser acessada em: https://rendleyarnou.github.io/PSMAR/. Os modelos de documentos utilizados neste processo podem ser vistos nos apêndices deste documento.

5.2 Fluxo Principal

O processo inicia quando surge a intenção de desenvolver um SMA com agentes racionais. O processo principal é composto por 5 subprocessos e 3 tarefas. A Figura 9 apresenta uma visão geral do PSMAR em *BPMN*.

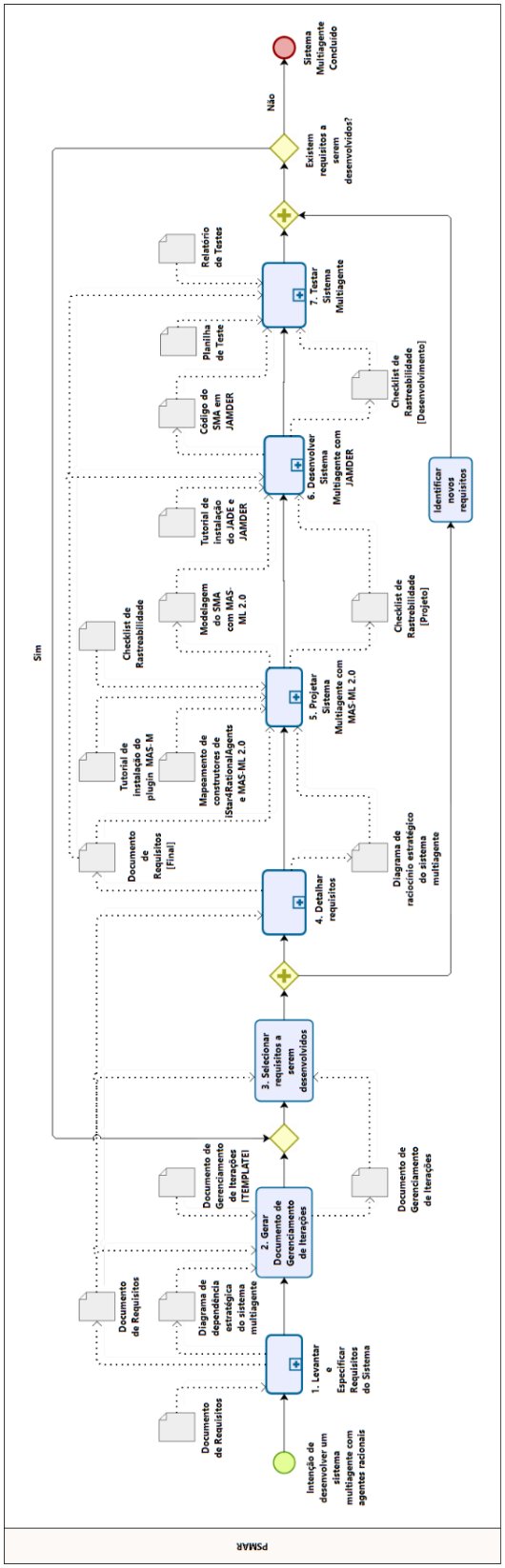
Os subprocessos do PSMAR encontram-se descritos a seguir:

1. **Levantar e especificar requisitos.** Este subprocesso consiste na identificação e descrição do SMA a nível de requisitos através do preenchimento do modelo de documento do documento de requisitos e da criação do modelo de requisitos utilizando iStar4RationalAgents.
2. **Gerar documento de gerenciamento de iterações.** Esta tarefa tem como finalidade criar o documento de gerenciamento de iterações do PSMAR.
3. **Selecionar requisitos a serem desenvolvidos.** Esta tarefa consiste em selecionar as entidades que serão desenvolvidas durante a iteração vigente do PSMAR.
4. **Detalhar requisitos.** Este subprocesso tem como finalidade detalhar os requisitos a serem desenvolvidos durante a iteração vigente do processo.
5. **Projetar sistema multiagente com MAS-ML 2.0.** Este subprocesso consiste na produção do modelo a nível de projeto através da modelagem do SMA com diagramas de atividade, diagrama de papéis e diagrama de organização utilizando a linguagem de modelagem MAS-ML 2.0. Esta tarefa é realizada utilizando o Eclipse com MAS-ML Tool.
6. **Desenvolver sistema multiagente com JAMDER.** Este subprocesso consiste na implementação do SMA através do framework JAMDER. Esta tarefa é realizada utilizando o Eclipse IDE for Java Developers e os frameworks JADE e JAMDER.
7. **Testar sistema multiagente.** Este subprocesso consiste no teste do SMA, identificação e correção de falhas. Nesta etapa, são realizados os testes das funcionalidades de

cada entidade do sistema. Caso seja identificada alguma falha, deverá ser corrigida.

8. **Identificar novos requisitos.** Esta tarefa tem como objetivo identificar novos requisitos durante cada iteração do processo.

Figura 9 - Fluxo principal do PSMAR

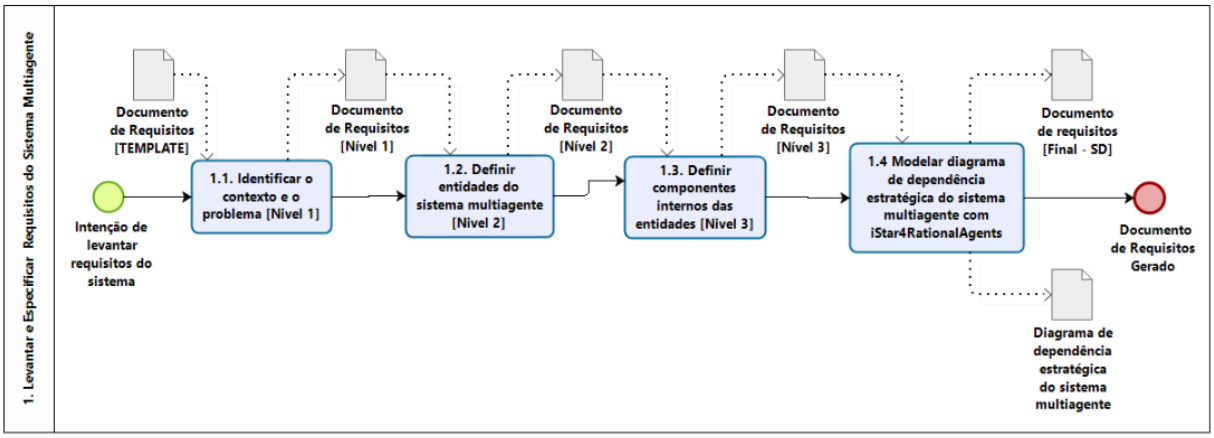


Fonte: Elaborada pelo autor.

5.3 Levantar e Especificar Requisitos do Sistema Multiagente

A execução deste subprocesso inicia com a necessidade de especificar os requisitos do sistema. Ele apresenta um conjunto de tarefas para apoiar o analista de requisitos a especificar os requisitos do SMA. O subprocesso pode ser visto na Figura 10 utilizando a notação *BPMN* (*Business Process Modeling Notation*) para representar. O subprocesso é detalhado nos parágrafos a seguir.

Figura 10 - Levantar e especificar requisitos do sistema multiagente



Fonte: Elaborada pelo autor.

O processo inicia a partir da intenção de levantar requisitos do SMA. A tarefa 1.1 Identificar o contexto e o problema, recebe o modelo de documento do documento de requisitos (disponível no Apêndice B) como artefato de entrada. Nesta etapa, o analista de requisitos deverá preencher o nível 1 de forma textual com intuito de descrever o sistema de forma geral.

O modelo de documento de requisitos foi adaptado utilizando como base o documento de requisitos disponível no site do Moodle da USP. (MOODLE USP, 2021).

Após finalizar a execução da tarefa 1.1 o nível 1 do documento estará preenchido, e então é iniciada a tarefa 1.2 Definir entidades do sistema multiagente recebendo como artefato de entrada o documento com o nível 1 já preenchido. Nesta etapa, o analista de requisitos deverá preencher o nível 2 de forma textual por meio de utilização de tabelas do documento com intuito de descrever as entidades do sistema.

Com a tarefa 1.2 finalizada o nível 2 do documento estará preenchido, por consequência, dando início a tarefa 1.3 Definir os componentes internos das entidades recebendo como artefato de entrada o documento com o nível 2 já preenchido. Nesta etapa o analista de requisitos deverá preencher o nível 3 de forma textual por meio de utilização de

tabelas com intuito de descrever os elementos intencionais das entidades do sistema, detalhando assim a arquitetura interna de cada agente.

Em seguida é iniciada a última tarefa deste subprocesso, 1.4 Modelar diagrama de dependência estratégica do sistema multiagente com *iStar4RationalAgents*, recebendo como artefato de entrada o documento com o nível 3 já preenchido. Nesta etapa o analista de requisitos deverá modelar o sistema a nível de requisitos com base nos conceitos e regras da linguagem, utilizando a ferramenta *piStar4RationalAgents* (GONÇALVES et al. 2019), disponível em https://www.cin.ufpe.br/~ler/piStar4rationalagents/.

Ao fim da execução deste subprocesso o analista de requisitos terá gerado como artefato de saída o documento de requisitos com os 3 níveis preenchidos juntamente com a modelagem a nível de requisitos do SMA.

5.4 Gerar Documento de Gerenciamento de Iterações

Esta tarefa inicia após finalizar o subprocesso 1 onde os requisitos são levantados e especificados. Têm como finalidade preencher o documento de gerenciamento de iterações (Apêndice I) para gerenciar as iterações realizadas no PSMAR e cada atividade desempenhada durante cada iteração.

5.5 Selecionar Requisitos a Serem Desenvolvidos

Esta tarefa inicia após finalizar a tarefa 2 onde é gerado o documento de gerenciamento de iterações. Esta tarefa consiste em selecionar as entidades que serão desenvolvidas durante a iteração vigente do PSMAR.

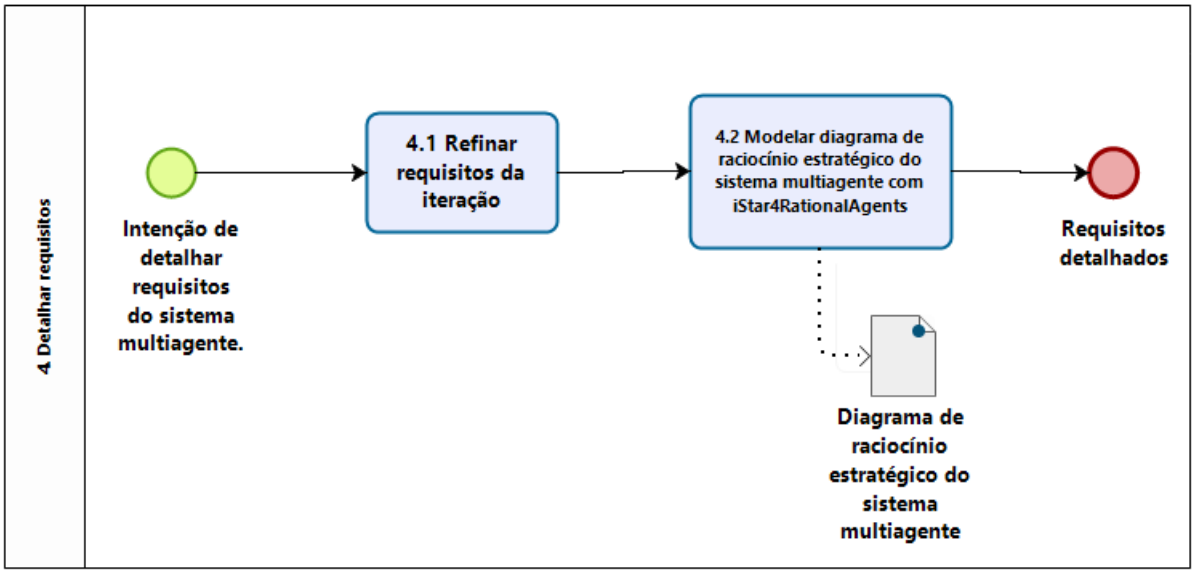
Durante as iterações do processo é indicado que seja seguida uma abordagem top-down, ou seja, deverão ser selecionadas primeiramente as entidades mais genéricas do sistema multiagente, como ambiente e organizações. Em seguida, os agentes e seus respectivos papéis devem ser selecionados considerando ordem decrescente de suas complexidades.

5.6 Detalhar Requisitos

Como mostra a Figura 11 esta tarefa inicia após finalizar a tarefa 3 onde são selecionados os requisitos a serem desenvolvidos. Esta tarefa consiste em detalhar os requisitos

a serem desenvolvidos durante a iteração vigente do processo. Gerando como artefato de saída o diagrama de raciocínio estratégico.

Figura 11 - Detalhar requisitos

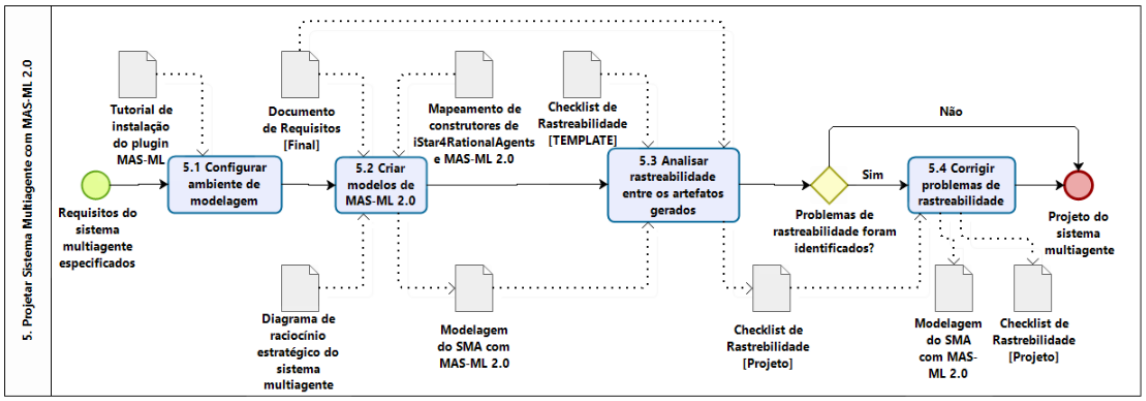


Fonte: Elaborada pelo autor.

5.7 Projetar Sistema Multiagente com MAS-ML 2.0

Este subprocesso é iniciado após o fim da execução do subprocesso 2. Ele apresenta um conjunto de tarefas para amparar o projetista a criar o projeto do SMA. O subprocesso pode ser visto na Figura 12 representado com a notação *BPMN* (*Business Process Modeling Notation*). O subprocesso é detalhado nos parágrafos a seguir.

Figura 12 - Projetar sistema multiagente com MAS-ML 2.0



Fonte: Elaborada pelo autor.

Após o fim do subprocesso 2, inicia-se o subprocesso 3 com a tarefa 5.1 Configurar ambiente de modelagem recebendo como artefato de entrada documento PDF de tutorial de

instalação do *plugin MAS-ML 2.0* (olhar o artefato em Apêndice G). Em seguida na tarefa 5.2 Criar modelos de *MAS-ML 2.0* recebe como artefatos de entrada o documento de requisitos final, o modelo de requisitos com *iStar4RationalAgents*, a planilha de mapeamento de construtores de *iStar4rationalAgents* e *MAS-ML 2.0* (olhar o artefato em Apêndice C). Os modelos de classe, papéis, organização, sequência e atividades serão gerados tomando como base o documento de requisitos e a modelagem de *iStar4RationalAgents* gerados no subprocesso 1, bem como considerando a planilha de mapeamento de construtores de *iStar4RationalAgents* e *MAS-ML 2.0*.

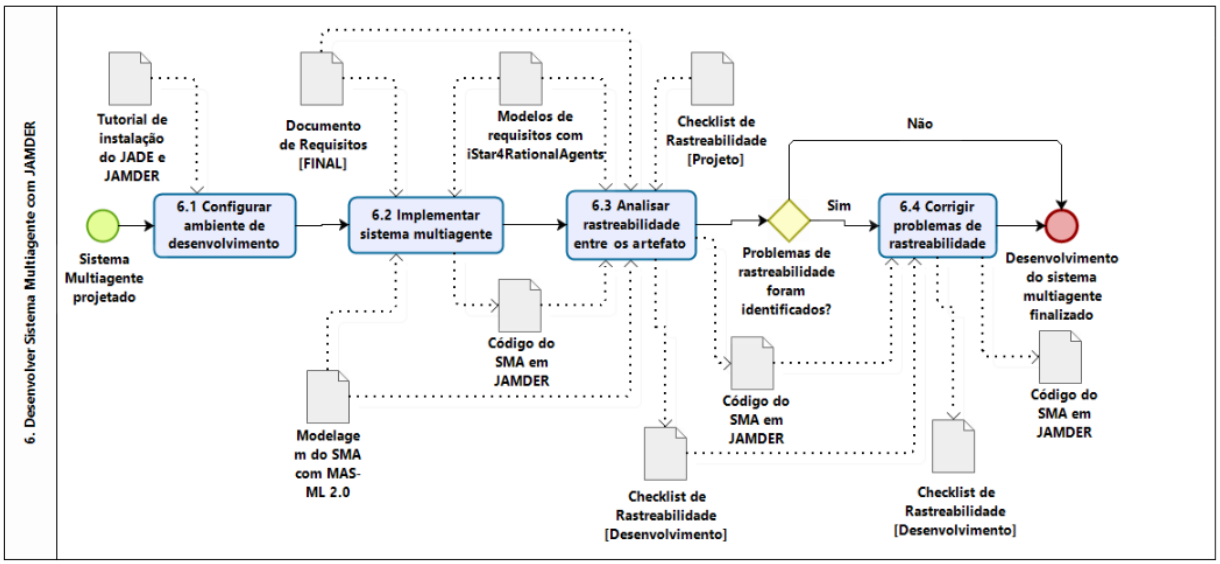
A modelagem do SMA com *MAS-ML 2.0* é gerada como artefato de saída pela tarefa 5.2 e é utilizada como entrada para a tarefa 5.3 Analisar rastreabilidade dos artefatos gerados. A tarefa 5.3 receberá como ainda um segundo artefato de entrada, trata-se do modelo de documento de checklist de rastreabilidade (disponível no Apêndice D). Nesta tarefa, os artefatos gerados pelo processo de desenvolvimento serão analisados com base no checklist de rastreabilidade.

Em seguida, é tomada uma decisão com base nas conclusões da análise de rastreabilidade. Caso problemas de rastreabilidade sejam identificados, a tarefa 5.4 Corrigir problemas de rastreabilidade é executada, recebendo o checklist de rastreabilidade como entrada. Se não, o subprocesso é finalizado com o projeto do SMA gerado.

5.8 Desenvolver Sistema Multiagente com JAMDER

Este subprocesso é iniciado após o fim da execução do subprocesso 3. Ele apresenta um conjunto de tarefas para amparar o programador a desenvolver o sistema multiagente. O subprocesso pode ser visto na Figura 13 O subprocesso é detalhado nos parágrafos a seguir.

Figura 13 - Desenvolver sistema multiagente com JAMDER



Fonte: Elaborada pelo autor.

O subprocesso 4 se inicia com a execução da tarefa 6.1 Configurar o ambiente de desenvolvimento, a qual recebe o tutorial de instalação do *JADE* e *JAMDER* (disponível no Apêndice D) como entrada. O desenvolvedor deve configurar o ambiente de desenvolvimento de acordo com este tutorial antes de iniciar a codificação do SMA. Em seguida, a tarefa 6.2 Implementar o sistema multiagente com *JAMDER* é executada, recebendo como artefatos de entrada o documento de requisitos final, o modelo de requisitos com *iStar4RationalAgents* e a modelagem do SMA em *MAS-ML 2.0*. A implementação deverá iniciar com a criação do código em *JAMDER* a partir do modelo de *MAS-ML 2.0*.

Em seguida na tarefa 6.3 Analisar rastreabilidade dos artefatos gerados que recebe como artefatos de entrada o documento de requisitos final, a modelagem do SMA com MAS-ML 2.0, o modelo de requisitos com iStar4RationalAgents, o código da implementação feito em JAMDER e o checklist de rastreabilidade com o nível de projetos preenchido, os artefatos são submetidos a análise através do *checklist* coma a finalidade de identificar possíveis inconsistências.

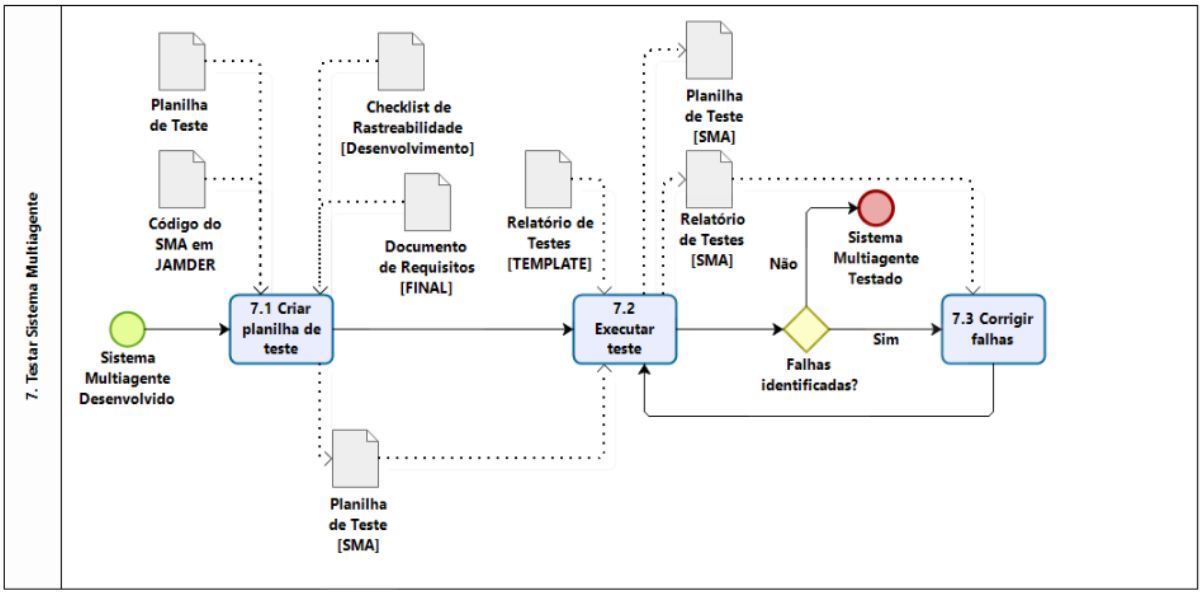
Em seguida é tomada uma decisão com base na existência de problemas de rastreabilidade. Caso problemas tenham sido identificados, a execução do subprocesso continua com a tarefa 6.4 Corrigir problemas de rastreabilidade que recebe o checklist com o nível de desenvolvimento preenchido. Nesta tarefa serão corrigidos os problemas identificados na tarefa 6.3 com base no checklist de rastreabilidade [Desenvolvimento] com o intuito de manter a consistência e rastreabilidades entre os artefatos gerados do projeto e a codificação do SMA.

Caso problemas de rastreabilidade não tenham sido identificados, o subprocesso é finalizado tendo gerado o código do SMA com *JAMDER*.

5.9 Testar Sistema Multiagente

Este subprocesso é iniciado após o fim da execução do subprocesso 4. Ele apresenta um conjunto de tarefas para auxiliar a realização de testes do SMA. O subprocesso pode ser visto na Figura 14 e é detalhado nos parágrafos a seguir.

Figura 14 - Testar sistema multiagente



Fonte: Elaborada pelo autor.

O fluxo de testes se inicia com o SMA desenvolvido, este subprocesso possui três tarefas. A tarefa 7.1 Criar planilha de teste consiste em preencher o modelo de documento da planilha de testes (disponível no Apêndice E) para o SMA em desenvolvimento. Ao fim desta tarefa será gerado o documento da planilha de testes [SMA] que irá servir com artefato de entrada para a tarefa 7.2 Executar teste. Na tarefa 7.2 serão executados os testes de cada funcionalidade com base na planilha de testes. Após a execução dos testes, será gerado o Relatório de Testes com base no modelo de documento de entrada (disponível no Apêndice F). O relatório de testes apresenta o resultado do teste de cada entidade do SMA.

Então é tomada uma decisão com base no relatório gerado como artefato de saída na tarefa 7.2. Caso sejam encontradas falhas descritas no relatório, a tarefa 7.3 Corrigir Falhas é iniciada recebendo como artefato de entrada o relatório, na tarefa 7.2. Nesta serão feitas as correções com base nas informações contidas no relatório de teste e então deverá retornar a

tarefa 7.2. Quando não, o processo é dado como finalizado, assim o sistema multiagente terá sido testado.

5.10 Identificar Novos requisitos

Esta tarefa acontece em paralelo entre os processos 2 e 5. Esta tarefa tem como objetivo identificar novos requisitos durante cada iteração do processo. Sempre que novos requisitos são identificados estes devem ser incluídos no Documento de Requisitos de modo a ser considerado nas próximas iterações.

6 ILUSTRAÇÃO DO USO DO PSMAR PARA DESENVOLVIMENTO DE SMA PARA O MOODLE

Nesta Seção serão apresentados os resultados da ilustração de uso do PSMAR.

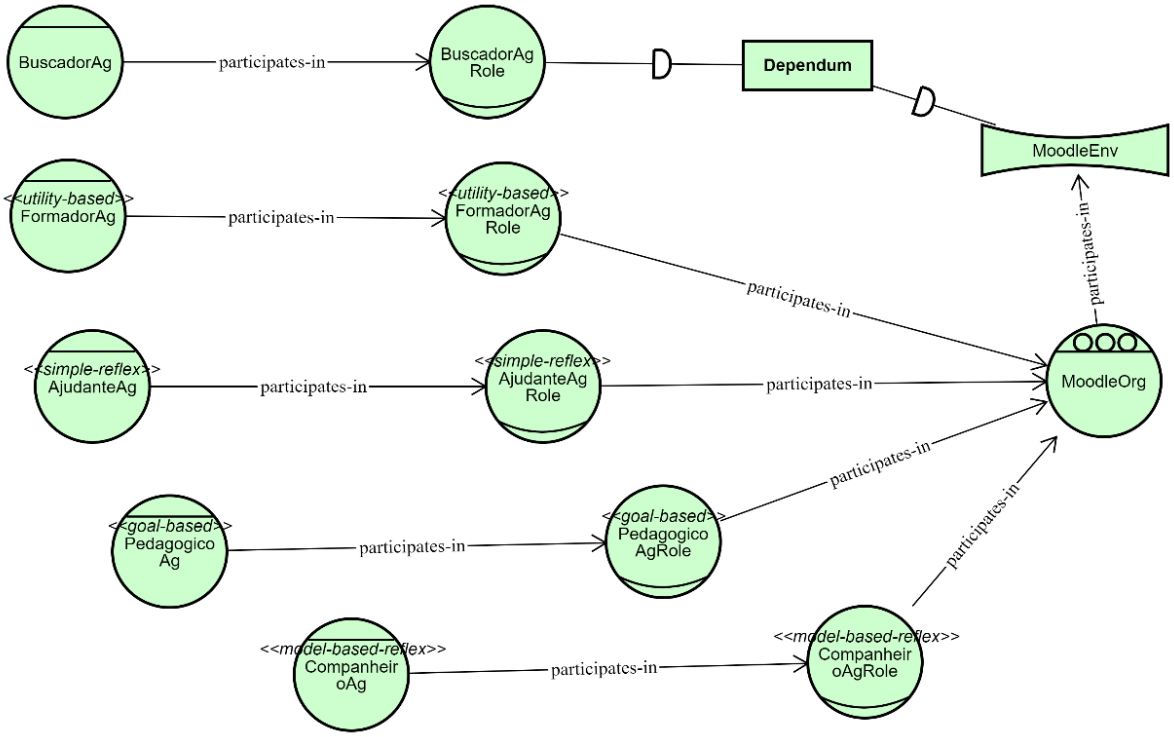
A ilustração do uso do PSMAR foi realizada por meio do desenvolvimento de um SMA para o MOODLE.

Todos os artefatos gerados durante o desenvolvimento com o PSMAR para este SMA do MOODLE estão disponíveis em https://github.com/rendleyarnou/tcc-ii-psmar-cs/tree/main/PSMAR%20MOODLE%20CASE%20STUDY/Moodle%20Docs. Neste repositório estão disponíveis os seguintes artefatos: documento de requisitos, modelos de iStar4RationalAgents, modelos de MAS-ML 2.0, código JAMDER, planilha de testes, relatório de testes, checklist de rastreabilidade e documento de gerenciamento das iterações.

Nesta seção apresentamos parte destes artefatos.

Na Figura 15 é apresentada a modelagem do diagrama de dependência estratégica utilizando a extensão *iStar4RationalAgents* contendo as entidades do sistema multiagente e seus relacionamentos. Esta modelagem é feita na etapa de levantamento de requisitos, após serem identificados o contexto e problema, definidos as entidades do sistema multiagente, assim como seus componentes internos. O artefato de documento de requisitos é utilizado como base para a criação da modelagem.

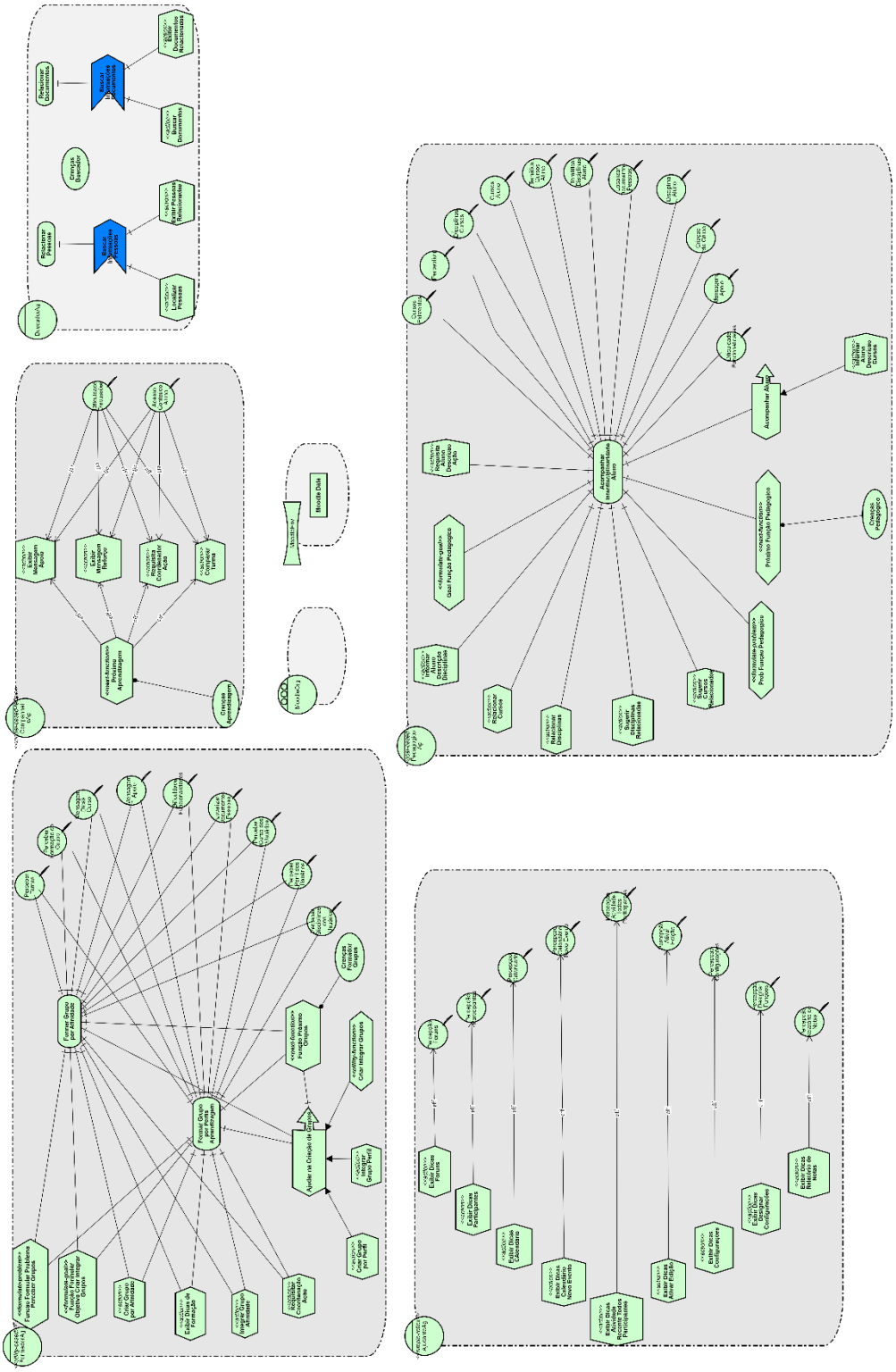
Figura 15 - Diagrama de dependência estratégica do MOODLE



Fonte: Elaborada pelo autor.

A Figura 16 contém a modelagem do diagrama de raciocínio estratégico também utilizando a extensão *iStar4RationalAgents,* contendo os componentes internos de cada entidade definida. Esta modelagem é realizada após o refinamento dos requisitos para que sejam especificados de forma mais detalhada. O artefato de documento de requisitos e a modelagem de dependência estratégica são utilizados como base para a modelagem do diagrama de raciocínio estratégico.

Figura 16 - Diagrama de raciocínio estratégico do MOODLE

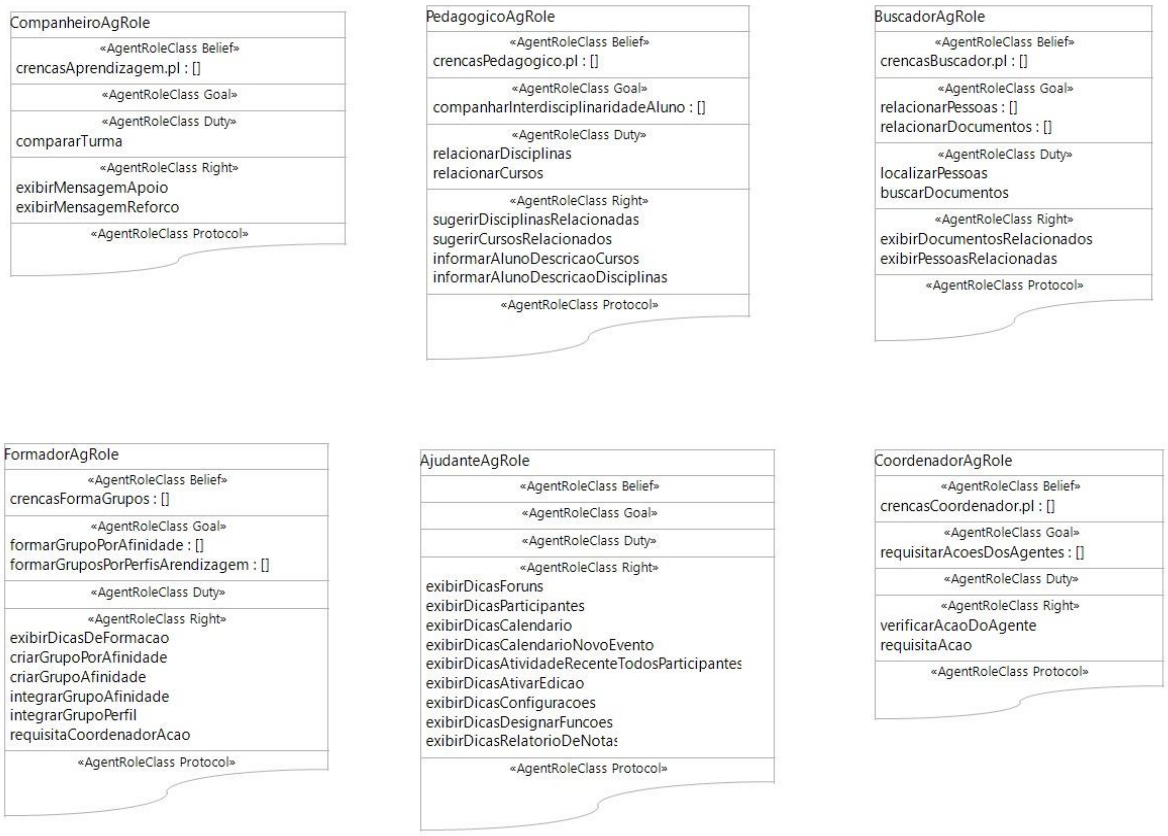


Fonte: Elaborada pelo autor.

As imagens a seguir são modelagens utilizando *MAS-ML 2.0*, esta modelagem é feita após a configuração do ambiente de modelagem seguindo como base o tutorial de instalação do plugin *MAS-ML Tool*. Os modelos de *MAS-ML 2.0* são gerados tomando como base o documento de requisitos e a modelagem de *iStar4RationalAgents*, bem como considerando a planilha de mapeamento de construtores de *iStar4RationalAgents* e *MAS-ML 2.0*.

A Figura 17 contém o diagrama de papéis dos agentes: Companheiro, Pedagógico, Buscador, Formador, Ajudante e Coordenador. Cada papel de agente possui os campos: crenças, objetivos, deveres, direitos e protocolo, com suas respectivas propriedades.

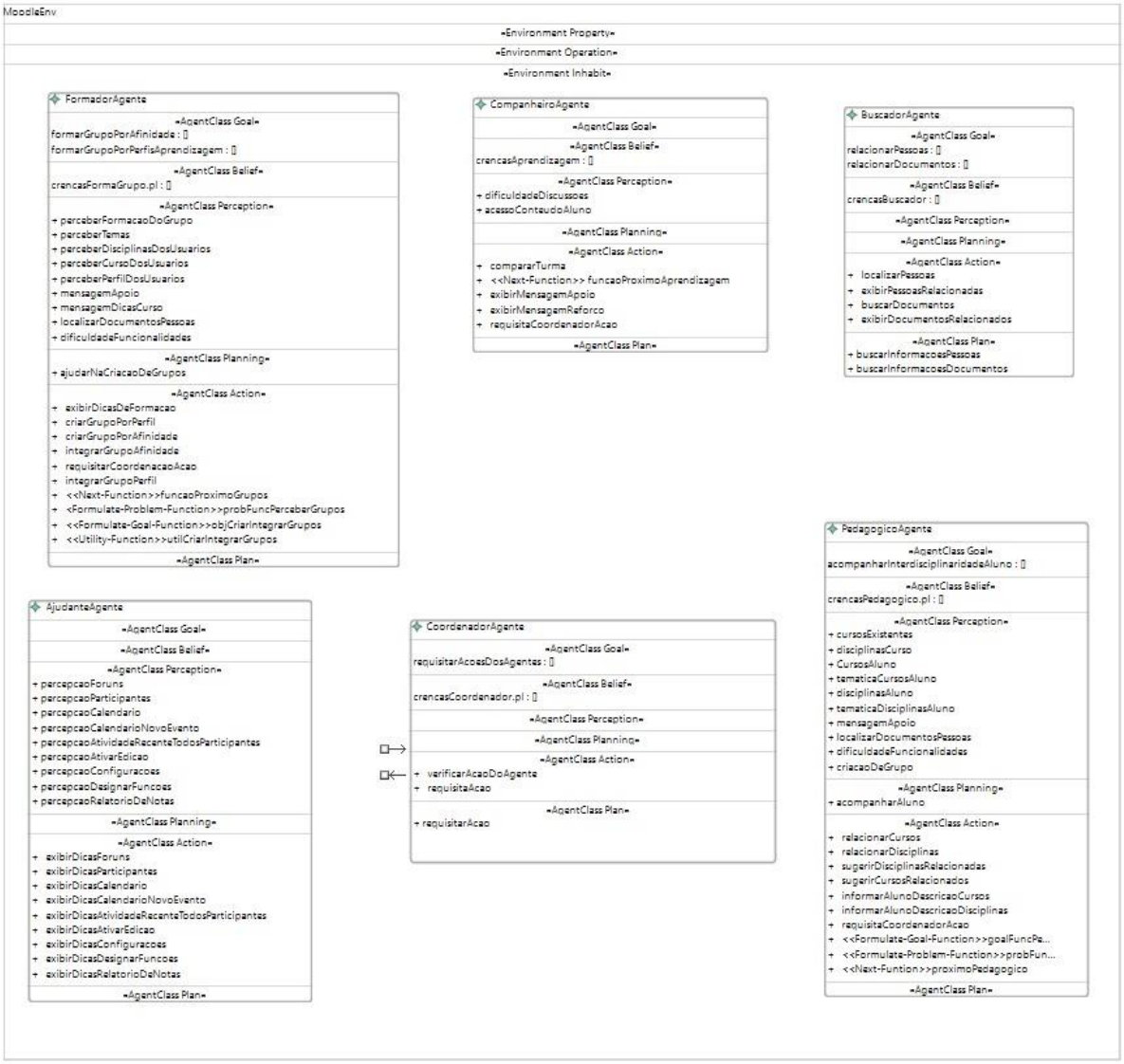
Figura 17 - Diagrama de papéis de MAS-ML 2.0 do MOODLE



Fonte: Elaborada pelo autor.

A Figura 18 contém o diagrama de classes contendo os mesmos agentes citados anteriormente habitando o ambiente MoodleEnv. Cada classe de agentes possui os campos: objetivos, crenças, percepções, planejamentos, ações e planos, contendo suas respectivas propriedades.

Figura 18 - Diagrama de classes de MAS-ML 2.0 do MOODLE



Fonte: Elaborada pelo autor.

A Figura 19 mostra a classe MoodleEnv que estende de Ambiente do *framework JAMDER*. No início do código é possível visualizar a importação do ambiente e da organização, bem como as classes dos agentes que estendem dos papéis e as classes dos agentes que estendem dos modelos, ambos de *JAMDER*.

Em seguida cada agente é instanciado no construtor da classe MoodleEnv e posteriormente a classe MoodleEnv é instanciada dentro da função *Main* do arquivo.

Figura 19 - Classe MoodleEnv desenvolvida com o framework JAMDER

```
import jamder.Environment;
import jamder.Organization;
import jamder.agents.GenericAgent;
import jamder.roles.AgentRole;
import jamder.roles.ModelAgentRole;
import jamder.roles.ProactiveAgentRole;
import moodle.AjudanteAgente;
import moodle.BuscadorAgente;
import moodle.CompanheiroAgente;
import moodle.CoordenadorAgente;
import moodle.FormadorAgente;
import moodle.PedagogicoAgente;

public class MoodleEnv extends Environment {
  public MoodleEnv(String name, String host, String port) {
    super(name, host, port);
    Organization MoodleOrg = new Organization("MoodleOrg", this, null);
    addOrganization("MoodleOrg", MoodleOrg);

    GenericAgent AjudanteAg = new AjudanteAgente("AjudanteAg", this, null);
    AgentRole AjudanteAgRole = new AgentRole("AjudanteAgRole", MoodleOrg,
    AjudanteAg);
    addAgent("AjudanteAg", AjudanteAg);

    GenericAgent BuscadorAg = new BuscadorAgente("BuscadorAg", this, null);
    AgentRole BuscadorAgRole = new ProactiveAgentRole("BuscadorAgRole", MoodleOrg,
    BuscadorAg);
    addAgent("BuscadorAg", BuscadorAg);

    GenericAgent CompanheiroAg = new CompanheiroAgente("CompanheiroAg", this, null);
    AgentRole CompanheiroAgRole = new ModelAgentRole("CompanheiroAgRole", MoodleOrg,
    CompanheiroAg);
    addAgent("CompanheiroAg", CompanheiroAg);

    GenericAgent CoordenadorAg = new CoordenadorAgente("CoordenadorAg", this, null);
    AgentRole CoordenadorAgRole = new ProactiveAgentRole("CoordenadorAg", MoodleOrg,
    CoordenadorAg);
    addAgent("CoordenadorAg", CoordenadorAg);

    GenericAgent FormadorAg = new FormadorAgente("FormadorAg", this, null);
    AgentRole FormadorAgRole = new ProactiveAgentRole("FormadorAgRole", MoodleOrg,
    FormadorAg);
    addAgent("FormadorAg", FormadorAg);

    GenericAgent PedagogicoAg = new PedagogicoAgente("PedagogicoAg", this, null);
    AgentRole PedagogicoAgRole = new ProactiveAgentRole("PedagogicoAgRole",
    MoodleOrg, PedagogicoAg);
    addAgent("PedagogicoAg", PedagogicoAg);
  }

  public static void main(String args[]) {
    MoodleEnv env = new MoodleEnv("MoodleEnv", "localhost", "8888");
    System.out.println("Executou");
  }
  // Additional attributes

  // Additional methods
}
```

Fonte: Elaborada pelo autor.

7 AVALIAÇÃO DO PSMAR

Esta seção apresenta os resultados da avaliação do PSMAR. O objetivo desta avaliação é identificar a percepção dos pesquisadores sobre o PSMAR.

Wohlin et al. (2012) define duas categorias principais de avaliação, ou seja, avaliações dinâmicas e estáticas. A avaliação estática não exige que a nova solução seja usada, ela pode ser feita por meio de uma apresentação da solução candidata seguida da análise. Já a avaliação dinâmica é uma abordagem em que a nova solução é utilizada em um projeto e os resultados são analisados, por meio de um estudo de caso, por exemplo. Neste estudo optamos por uma avaliação estática do PSMAR.

Este estudo é baseado em questionário (KITCHENHAM, B.; PFLEEGER, 2002) e foi realizado com pesquisadores e desenvolvedores que atuam no desenvolvimento de sistemas multiagentes.

As próximas subseções apresentam a caracterização deste estudo.

7.1 Universo e amostra dos participantes

O universo desta pesquisa é composto por pesquisadores e desenvolvedores da área de sistemas multiagentes. Consideramos como amostra, os autores de artigos das últimas 5 edições dos eventos WESAAC (Workshop-School on Agents, Environments, and Applications) e BRACIS (Brazilian Conference on Intelligent Systems). Estas conferências foram selecionadas devido a relevância na comunidade brasileira de SMA. Assim, a amostra da pesquisa consiste de 264 autores de artigos destas conferências.

7.2 Preparação para coleta

O questionário foi estruturado por meio de um formulário com um conjunto de questões de múltipla escolha e questões abertas. As questões possuem opções definidas com base na escala Likert com os seguintes valores: Concordo plenamente (1), Concordo (2), Não sei (3), Discordo (4), Discordo plenamente (5).

Os roteiros da pesquisa foram validados por um pesquisador com experiência na área de SMA. O questionário também foi testado anteriormente por um piloto que ajudou a melhorá-lo.

O questionário conta inicialmente com um vídeo sobre o PSMAR, o qual é exibido antes das perguntas. Este vídeo está disponível em https://youtu.be/crNipI9OQ04.

O questionário possui 17 questões de múltipla escolha e 4 questões abertas. A estrutura do questionário está disponível no Apêndice A. As questões visam coletar a opinião dos participantes sobre a necessidade de se propor um processo para apoiar o desenvolvimento de SMA com agentes racionais, aspectos específicos como a especificação textual, o gerenciamento de iterações, modelos, testes, rastreabilidade, nível de dificuldade e viabilidade de uso e relevância e utilidade do PSMAR.

7.3 Coleta dos dados

A pesquisa foi conduzida em português, submetida pelos formulários do Google e direcionadas aos participantes via e-mail durante os meses de Agosto de 2021 a Novembro de 2022.

Ao final deste período, obtivemos 29 respostas de participantes de 18 instituições diferentes que concordaram em participar da pesquisa.

A distribuição das instituições dos participantes distribui-se da seguinte forma: 1 do Centro de Pesquisa de Ensino Tecnológico, 1 da Empresa Baiana de Água e Saneamento, 1 do Instituto Federal do Ceará, 1 não possuem instituição de trabalho/estudo, 3 da Pontifícia Universidade Católica do Rio Grande do Sul, 1 da Prefeitura Municipal de São Leopoldo, 5 da Universidade Federal do Ceará, 1 da Universidade do Estado de Santa Catarina, 2 da Universidade Federal Rural do Semi-Árido, 2 da Universidade de São Paulo, 2 da Universidade Estadual Vale do Acaraú, 1 da Ubivis IoT Otimização Industrial, 2 da Universidade Federal de Santa Catarina, 2 da Universidade do Rio Grande, 1 da CWI Digital House, 1 da Petrobras, 1 da Weni e 1 atua com representação/modelagem de campos sociais.

No que se refere ao grau de escolaridade dos participantes, temos 1 aluno de graduação, 2 graduados, 4 alunos de mestrado, 6 mestres, 11 alunos de doutorado, 5 doutores.

Os cargos ocupados pelos participantes são: 10 professores, 13 alunos, 1 desenvolvedor júnior, 1 desenvolvedora back-end, 1 engenheiro sênior, 1 analista de TIC, 1 analista de sistemas e 1 não leciona.

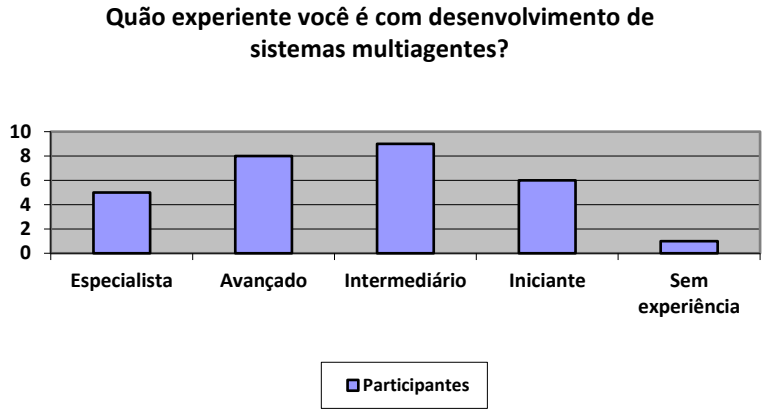
A experiência com o desenvolvimento de sistemas multiagentes é resumida da seguinte forma: 1 sem experiência, 6 iniciantes, 9 intermediários, 8 avançados e 5 especialistas.

7.4 Resultados da avaliação

Esta seção explana os resultados e discussões, estruturada de forma que aponta o entendimento dos participantes a respeito da aplicação e prestabilidade do PSMAR.

No que diz respeito ao nível de experiência dos participantes da pesquisa é possível identificar que a quantidade de participantes que se identificam com nível de conhecimento intermediário e avançado (9 e 8 respectivamente) é bem maior que a porcentagem que se identifica com nível especialista (5), iniciante (6) e sem experiência (apenas 1) como mostrado na Figura 20.

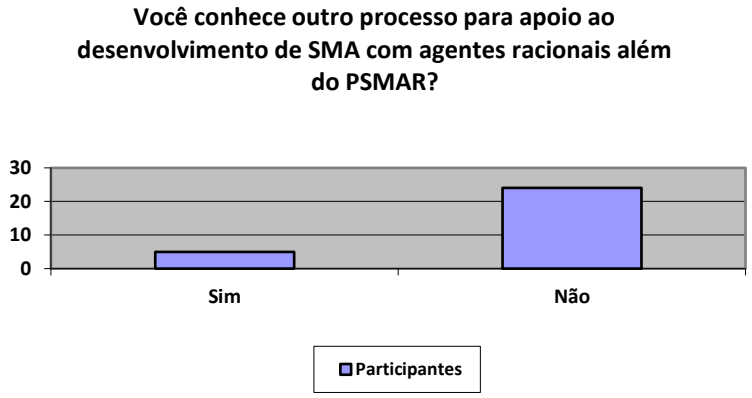
Figura 20 - Nível de experiência dos participantes com desenvolvimento de sistemas multiagentes



Fonte: Elaborada pelo autor.

Para o questionamento se conhecem outro processo de apoio ao desenvolvimento de sistemas multiagentes além do PSMAR, a Figura 21 mostra que aproximadamente 83% dos participantes afirmam não conhecer e apenas 17% afirmam que conhecem um processo.

Figura 21 - Quantidade de participantes que conhecem outro processo para desenvolvimento de sistemas multiagentes com agentes racionais



Fonte: Elaborada pelo autor.

Na sequência da pesquisa, foi disponibilizada uma caixa de texto para que os participantes que afirmaram conhecer um processo pudessem informar a respeito do mesmo.

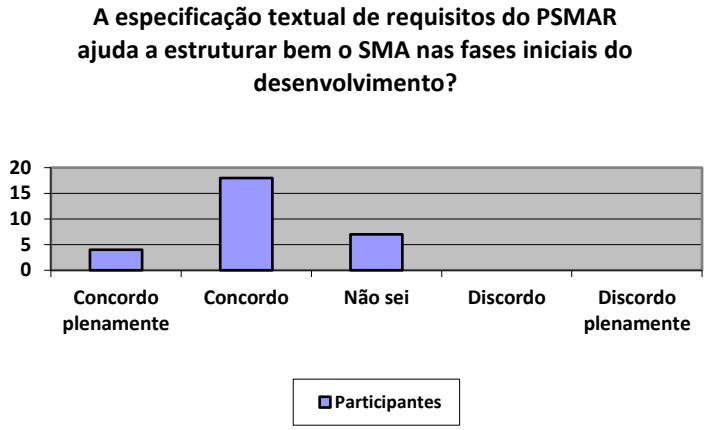
Os participantes informaram técnicas que apoiam parte das etapas do ciclo de desenvolvimento, como JaCaMo e JIAC, que são frameworks que apoiam a atividade de codificação do SMA, e Prometheus, que apoia a especificação modelos.

Para as questões a seguir possuem opções definidas com base na escala Likert com valores descritos anteriormente como mostram as imagens da Figura 22 a Figura 29.

Figura 22 - É necessária a proposição de um processo para conduzir o desenvolvimento de SMAs com agentes racionais

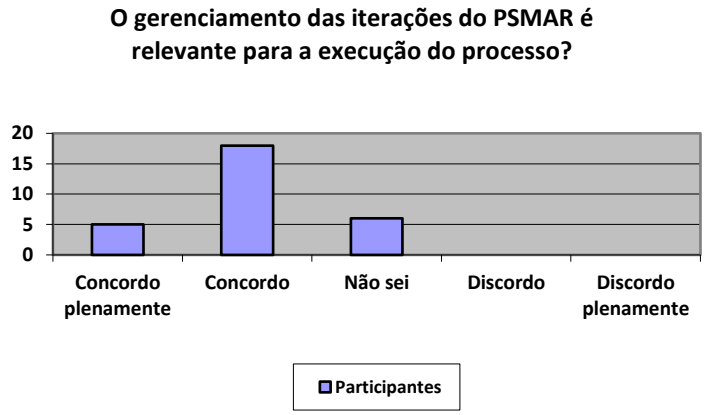
Fonte: Elaborada pelo autor.

Figura 23 - A especificação textual de requisitos do PSMAR ajuda a estruturar bem o SMA nas fases iniciais do desenvolvimento



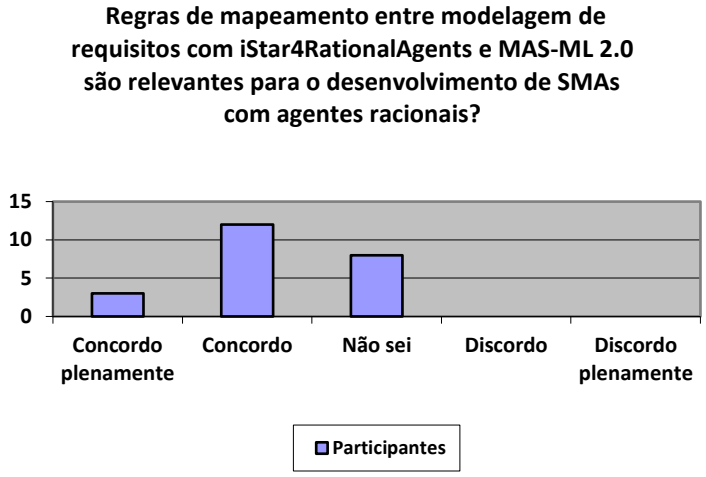
Fonte: Elaborada pelo autor.

Figura 24 - O gerenciamento das iterações do PSMAR é relevante para a execução do processo



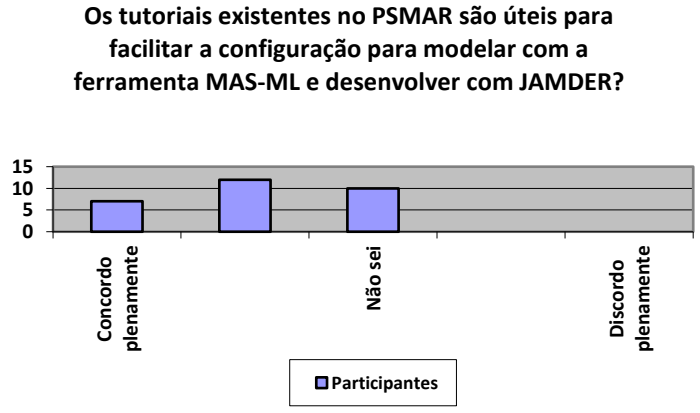
Fonte: Elaborada pelo autor.

Figura 25 - Regras de mapeamento entre modelagem de requisitos com iStar4RationalAgents e MAS-ML 2.0 são relevantes para o desenvolvimento de SMAs com agentes racionais



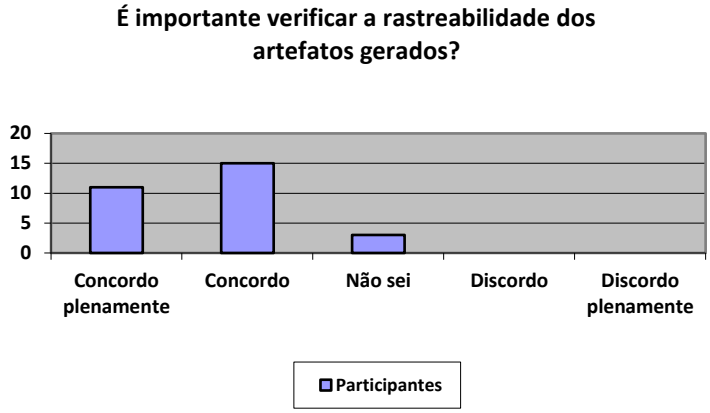
Fonte: Elaborada pelo autor.

Figura 26 - Os tutoriais existentes no PSMAR são úteis para facilitar a configuração para modelar com a ferramenta MAS-ML e desenvolver com JAMDER



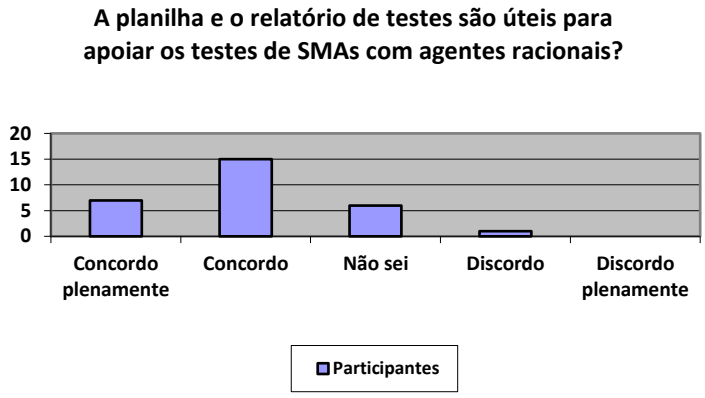
Fonte: Elaborada pelo autor.

Figura 27 - É importante verificar a rastreabilidade dos artefatos gerados



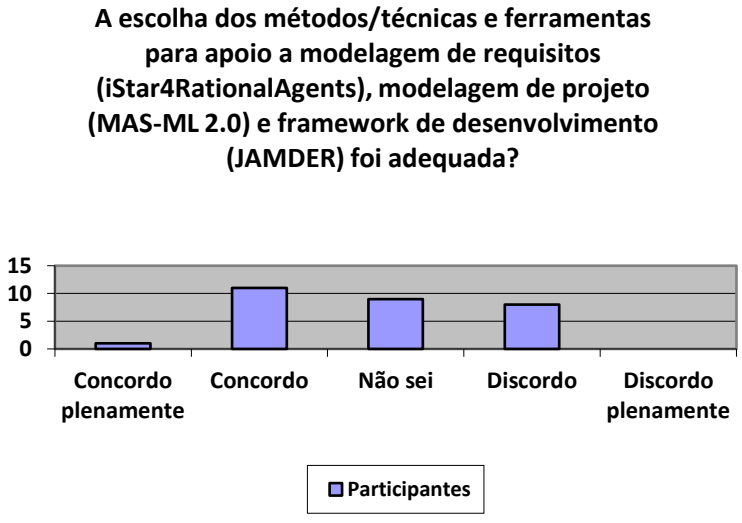
Fonte: Elaborada pelo autor.

Figura 28 - A planilha e o relatório de testes são úteis para apoiar os testes de SMAs com agentes racionais



Fonte: Elaborada pelo autor.

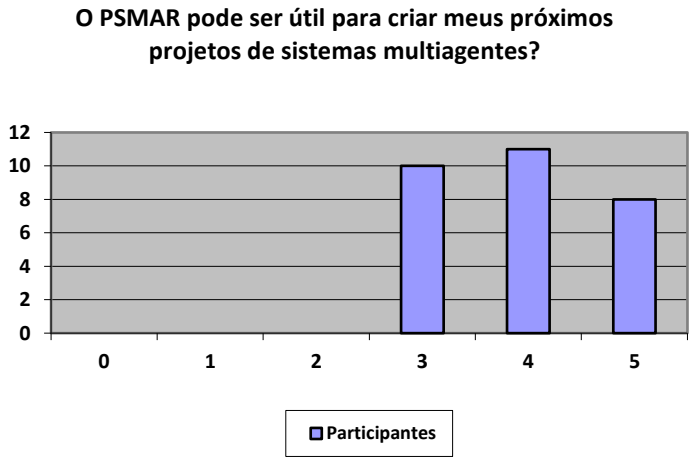
Figura 29 - A escolha dos métodos/técnicas e ferramentas para apoio a modelagem de requisitos (iStar4RationalAgents), modelagem de projeto (MAS-ML 2.0) e framework de desenvolvimento (JAMDER) foi adequada



Fonte: Elaborada pelo autor.

A Figura 30 se refere a utilidade do PSMAR e possuí opções com valores de 0 a 5 onde 0 indica ‘Não contribui’ e 5 ‘Contribui muito’

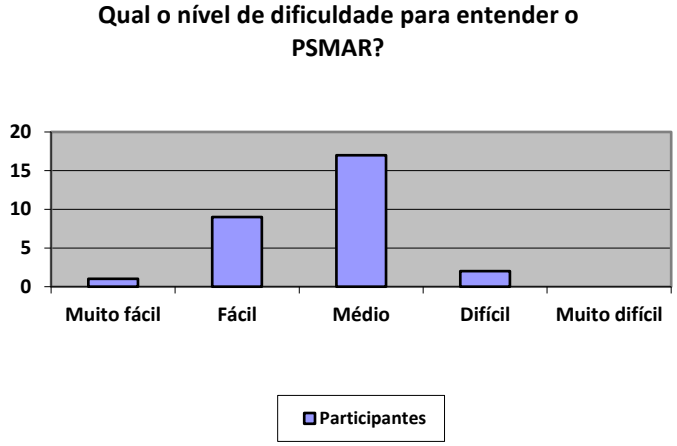
Figura 30 - O PSMAR pode ser útil para criar meus próximos projetos de sistemas multiagentes



Fonte: Elaborada pelo autor.

A respeito do nível de dificuldade os participantes responderam da seguinte forma como mostra a Figura 31:

Figura 31 - Qual o nível de dificuldade para entender o PSMAR



Fonte: Elaborada pelo autor.

Parte dos participantes destacou a dificuldade em analisar esta questão por conta de não terem utilizado o processo de fato. Assim sendo, estes informaram que a opinião se baseou somente no vídeo de apresentação do processo. Outro ponto importante destacado foi a dificuldade de entender o processo por não ter conhecimento suficiente em parte das técnicas envolvidas como iStar4RationalAgents, MAS-ML ou JAMDER, por exemplo.

Sobre o uso do PSMAR ser viável para criação de SMAs com agentes racionais, adequação, utilidade e finalidade de uso, as tabelas 2, 3, 4 e 5 apresentam estes resultados relacionados.

Tabela 2 - Utilidade do PSMAR para a criação de sistemas multiagentes com agentes racionais

Questões	Concordo plenamente	Concordo	Não sei	Discordo	Discordo plenamente
O uso do PSMAR é viável para a criação de SMAs com agentes racionais	3	20	6	0	0
O PSMAR pode ser útil para o desenvolvimento de SMAs pelos desenvolvedores e pesquisadores da área	7	20	2	0	0
O PSMAR pode ser útil para criar meus próximos projetos de SMAs	5	15	6	3	0

Fonte: Elaborada pelo autor.

Tabela 3 - Adequação do PSMAR para futuros projetos de sistemas multiagentes com agentes racionais

Questão	Muito adequado	Adequado	Não sei	Inadequado	Muito inadequado
Quão adequado é o uso do PSMAR na definição dos futuros projetos de sistemas multiagentes	3	15	11	0	0

Fonte: Elaborada pelo autor.

Tabela 4 - Utilidade do PSMAR para níveis de experiência de desenvolvedores

A quem o PSMAR pode ser útil	Desenvolvedores sem experiência no desenvolvimento de SMAs com agentes racionais	Desenvolvedores com experiência no desenvolvimento de SMAs com agentes racionais	Ambos	Ninguém
Quantidade de votos dos participantes	5	5	19	0

Fonte: Elaborada pelo autor.

Tabela 5 - Uso da proposta do PSMAR

Vou usar PSMAR como foi proposto	Vou usar grande parte das tarefas e artefatos do PSMAR como foi proposto e fazer algumas adaptações	Vou usar algumas tarefas e artefatos de PSMAR para criar minha instância deste processo	Não vou usar nada do PSMAR
2	8	8	3

Fonte: Elaborada pelo autor.

É possível perceber pela distribuição das respostas que os participantes consideram o PSMAR viável, adequado e útil para o desenvolvimento de SMA com agentes racionais.

Questionados se recomendariam o PSMAR a outros desenvolvedores, 20 participantes responderam que recomendariam, e apenas 9 afirmaram que não recomendariam.

Sobre os pontos fortes do PSMAR, 4 participantes destacaram o uso de ferramentas e frameworks conhecidos, como apresentado pelo participante 8: *" (PSMAR) Está apoiado em frameworks, ferramentas e notações muito difundidas na área de engenharia de software e MAS."*. De maneira complementar, 7 participantes mencionaram a organização que o processo impõe ao desenvolvimento, como mencionado pelo participante 16: *"A organização que o próprio processo impõe é um ponto muito forte, juntamente com a rastreabilidade de requisitos de ponta a ponta até a etapa de testes."*. Além disto, os participantes também destacaram como pontos positivos a facilidade no entendimento, o fato do processo apresentar de maneira detalhada tarefas e artefatos, além de ser modelado em BPMN, uma notação amplamente conhecida e utilizada para representar processos.

Por outro lado, sobre os pontos fracos do PSMAR, 8 participantes destacaram a quantidade de tarefas (tamanho) do processo como um possível entrave na adoção do processo. De maneira complementar, o participante 2 menciona que o processo pode não ser adequado aos projetos de pequeno e médio porte: *"Para pequenos e médios projetos *parece* sobrecarregar a equipe com documentação excessiva"*. Os participantes 1 e 19 destacaram que o processo é poderia ser mais abrangente caso fosse atrelado a técnicas específicas: *"O processo proposto depende do uso de ferramentas e protocolos específicos. Um processo "agnóstico" teria um alcance maior."* Participante 1. *"Acredito que a etapa de desenvolvimento poderia ser mais genérica, atendendo outras ferramentas de desenvolvimento, sobretudo sistemas de simulação baseada em agentes."* Participante 19.

Por último, questionamos sobre melhorias identificadas. Os principais aspectos citados foram: gerar uma versão mais enxuta do PSMAR, adequada a projetos de pequeno e médio porte. A adequação do processo a cenários mais específicos como simulação também foi mencionada. Além disto, 4 participantes mencionaram a necessidade de proposição de ferramentas de apoio a especificação de requisitos textuais, rastreabilidade, testes, dentre outros.

7.5 Ameaças a validade

Esta seção discute sobre as ameaças a validade deste estudo. Para (KITCHENHAM; PFLEEGER, 2002) 4 (quatro) aspectos devem ser considerados: Validade de Face, Validade de Conteúdo, Validade de Conclusão, Validade de Construção.

7.5.1 Validade de face

Pode ser entendida como uma revisão superficial dos itens do instrumento por pessoas inexperientes. O questionário feito no Google Forms foi apresentado para 10 alunos de graduação da Universidade Federal do Ceará – Campus Quixadá que não tinham conhecimento no assunto avaliado pelo survey, com o objetivo de revisar a estrutura, o design e a objetividade das tarefas do questionário. Mesmo sem conhecimentos prévios sobre os alunos apresentaram um bom entendimento das perguntas.

7.5.2 Validade de conteúdo

Trata-se de uma avaliação subjetiva de como o instrumento parece adequado a um grupo de pessoas com conhecimento sobre o assunto. Realizamos um piloto envolvendo 10 pessoas, sendo uma delas especialista em desenvolvimento de SMA O objetivo foi testar a compreensão dos participantes sobre a pesquisa e assegurar que ela inclui todas as demandas necessárias para a avaliação.

7.5.3 Validade de conclusão

Diz respeito a habilidade de alcançar o objetivo correto sobre os dados coletados, utilizando testes estatísticos, e quão confiáveis são as medidas e esses dados. Devido a um não tão alto número de participantes neste estudo, não poderíamos fazer inferências estatísticas sobre os dados o que gerou uma ameaça a validade deste estudo.

7.5.4 Validade de construção

É a observação de como o instrumento de pesquisa porta-se quando está em uso. Os dados obtidos podem ser convergentes ou divergentes a depender de problemas que possam surgir, sejam por falhas do pesquisador ou dos participantes, por exemplo: Os participantes podem basear seu comportamento em suposições. O ser humano geralmente tenta parecer melhor do que é quando está sendo avaliado. Os pesquisadores podem projetar o experimento pensando nos resultados que esperam (viés).

Para tentar mitigar essa ameaça buscamos elaborar questões objetivas, que foram validadas nos testes piloto que foram realizados.

8 CONCLUSÃO

Nesta monografia foi apresentado o PSMAR como resultado da união dos trabalhos de *iStar4RationalAgents*, *MAS-ML 2.0* e *JAMDER* tomando como base o estudo de modelos de processos de *software* sob determinada perspectiva com intenção de viabilizar a criação do PSMAR fundamentado nas variações desses processos para estabelecer os estágios que definem as atividades fundamentais de desenvolvimento.

Dessa forma o trabalho estabeleceu atividades para sistematizar a especificação textual dos requisitos e gerou artefatos visando contribuir para o uso do processo. Foi criado o Documento de Requisitos (Apêndice B) utilizando uma abordagem de especificação textual baseada em diferentes níveis de especificação. Também foi estabelecida a definição do mapeamento entre *iStar4RationalAgents* e *MAS-ML 2.0,* gerando a Planilha de Mapeamento de Construtores iStar4RationalAgents e *MAS-ML 2.0* (Apêndice C) que evidencia o mapeamento entre as entidades com o propósito de facilitar a conversão entre os modelos. O Checklist de Rastreabilidade (Apêndice D) dispõe-se a analisar a rastreabilidade entre as fases do PSMAR com o propósito de checar a conformidade do sistema multiagente. O Tutorial de Instalação do *JADE* e *JAMDER* (Apêndice E) auxilia a configuração de ambiente de desenvolvimento, da mesma maneira que o Tutorial de instalação do plugin *MAS-ML Tool* (Apêndice H). Para a etapa de testes do sistema multiagente foram elaborados os modelos de documento de planilha de testes (Apêndice F) e relatório de testes (Apêndice G).

O PSMAR foi modelado de forma detalhada utilizando BPMN. Assim, o processo foi disponibilizado e está acessível para os pesquisadores e desenvolvedores de SMA que pretendam utilizar as técnicas nele contidas. O uso do processo foi ilustrado por meio do desenvolvimento de um SMA para o MOODLE, sendo este considerado adequado.

Após a criação e ilustração deste processo, foi realizada uma validação do PSMAR por meio de um questionário a fim de coletar a opinião de pesquisadores e desenvolvedores da área sobre aspectos como a necessidade de criação do PSMAR, dificuldade de entender e sua utilidade.

Como solução, foi possível constatar que o presente trabalho auxilia o desenvolvimento de sistemas multiagentes com agentes racionais pois dá suporte ao desenvolvimento provendo atividades bem descritas, além de artefatos e técnicas relacionadas.

Como trabalhos futuros visualizamos a possibilidade da criação de uma abordagem de transformação automática de modelos de *iStar* para modelos de *MAS-ML* e também a adequação do gerador de código do *MAS-ML 2.0* para *JAMDER*. O uso do processo por outro

pesquisador de modo a ser um estudo de caso do processo, com objetivo de trazer um melhor entendimento sobre o processo e insights para sua melhoria.

REFERÊNCIAS

BITTENCOURT, G. **Inteligência artificial distribuída.** I workshop de computação do ITA, Instituto Tecnológico de Aeronáutica. 1998.

CASTRO, J.; ALENCAR, F.; SILVA, C. T. L. L. Engenharia de software orientada a agentes. In: BREITMAN, K.; ANIDO, R. (Ed.). **Atualizações em Informática**. Rio de Janeiro: Editora PUC-Rio, 2006. p. 245-282

CHELLA, A.; COSSENTINO, M.; SEIDITA, V. Towards a methodology for designing artificial conscious robotic systems. 2009. In: BIOLOGICALLY INSPIRED COGNITIVE ARCHITETURES II. **Anais [...]**. Disponível em: http://www.aaai.org/ocs/index.php/FSS/FSS09/paper/view/882/1274. Acesso em: 7 out. 2020.

CRESWELL, J. **A Concise Introduction to Mixed Methods Research**. [S. l.] Sage Publications, 2014.

DALPIAZ, F.; FRANCH, X; HORKOFF, J. iStar 2.0 Language Guide. **arXiv:1605.07767**, 2016. Disponível em: https://arxiv.org/pdf/1605.07767v1.pdf. Acesso em: 7 out. 2020.

FRANKLIN, S.; GRAESSE, A. Is it an agent, or just a program? a taxonomy for autonomous agents. In: THIRD INTERNATIONAL WORKSHOP ON AGENTS THEORIES. [S.l.]: Springer- Verlag, 1996.

FREIRE, E. S.; GONÇALVES, E.; CORTÉS, M. I.; BRANDÃO, M. TAO+: Extending the Conceptual Framework TAO to Support Internal Agent Architectures in Normative Multi-Agent Systems. **Electronic Notes in Theoretical Computer Science**, v. 292, 2013.

GONÇALVES, E. J. T.; DE OLIVEIRA, M. A.; FREIRES JUNOIR, j. H.; FEITOSA, G. E. S.; MENDES, D. H.; CORTÉS, M. I.; FEITOSA, R. G. F.; LOPES, Y. S.; Uma Abordagem Baseada em Agentes de Apoio ao Ensino a Distância Utilizando Técnicas de Engenharia de Software. In: SIMPÓSIO BRASILEIRO DE SISTEMAS DE INFORMAÇÃO (SBSI), 10. 2014, Londrina. **Anais [...]**. Porto Alegre: Sociedade Brasileira de Computação, 2014. p. 219-230

GONÇALVES, E. J. T. ; **Modelagem de arquiteturas internas de agentes de software utilizando a linguagem MAS-ML 2.0**. 2009. 105 f. Dissertação (Mestrado Acadêmico ou Profissional em 2009) - Universidade Estadual do Ceará, , 2009. Disponível em: http://siduece.uece.br/siduece/trabalhoAcademicoPublico.jsf?id=57200. Acesso em: 12 dez. 2022

GONÇALVES, E.; ARAUJO, J.; CASTRO, J. iStar4RationalAgents: Modeling Requirements of Multi-Agent Systems with Rational Agents. In: **38th INTERNATIONAL CONFERENCE ON CONCEPTUAL MODELLING**, 2019. p. 558-566

GONÇALVES, E.; CORTÉS, M. I.; CAMPOS, G.; LOPES, Y.; FREIRE, E.; SILVA, V. T.; OLIVEIRA, K.; OLIVEIRA, M. A. MAS-ML 2.0: Supporting the Modelling of Multi-Agent Systems with Different Agent Architectures. **The Journal of Systems and Software**, v 108, pp. 77-109, 2015.

GONÇALVES, E. J. T. et al. Towards the modeling reactive and proactive agents by using mas-ml. In: SAC 10: PROCEEDINGS OF THE 2010 ACM SYMPOSIUM ON APPLIED COMPUTING. New York, NY, USA: ACM, 2010. p. 936–937. ISBN 978-1-60558-639-7.

HENDERSON-SELLERS, B. Creating a comprehensive agent-oriented methodology – using method engineering and the open metamodel. In: the OPEN metamodel, **Chapter 13 in Agent-Oriented Methodologies** (eds. B. Henderson-Sellers and P. Giorgini), Idea Group. [S.l.: s.n.], 2005. p. 11.

II, M. J. de O. M.; **Mas-Commonkads+**: Uma Extensão a Metodologia Mas-Commonkads para Suporte ao Projeto Detalhado de Sist. 2010. Sem Numeração Dissertação (Mestrado Acadêmico ou Profissional em 2010) - Universidade Estadual do Ceará, 2010. Disponível em: http://siduece.uece.br/siduece/trabalhoAcademicoPublico.jsf?id=67815. Acesso em: 10 dez. 2022

IGLESIAS, C. A.; GARIJO, M. The agent-oriented methodology mas-commonkads. In: GIORGINI, B. H.-S. e P. (Ed.). **Agented-Oriented Methodologies**. [S.l.]: IDEA Group Publishing, p. 46–78, 2005.

JADE. **Java Agent Development Framework**, [S.I: s.n] 2020. Disponível em: http://jade.tilab.com/. Acesso em: 7 out. 2020.

JENNINGS, N. R. Coordination techniques for dai. In: O'HARE, G.; JENNINGS, N. R. (Ed.). **Foundations of distributed artificial intelligence.** [S.l.]: John Wiley and Sons, 1996.

KITCHENHAM, B.; PFLEEGER, S. Principles of Survey Research, **Software Engineering Notes**, v. 27, n. 5, pp. 1- 20, 2002.

LOPES, Y. S.; CORTÉS, M. I.; GONÇALVES, E. J. T.; OLIVEIRA, R.; JAMDER: JADE to MULTI-Agent Systems Development Resource. **ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal**, 7(3), 63–98. Disponível em: https://doi.org/10.14201/ADCAIJ2018736398. Acesso em: 7 out. 2020.

MELO, J.; BRITO, A.; SA FILHO, C.; JUNIOR, J.; ALENCAR, F. Formalização de Regras de Mapeamento de i* para Diagrama de Classe em UML. **29th SIMPÓSIO BRASILEIRO DE ENGENHARIA DE SOFTWARE**, 2015.

MERRIAM, S. **Qualitative Research:** a guide to design and implementation, Jossey-Bass, 2009.

MICHAELIS. **Agente**. Disponível em: https://michaelis.uol.com.br/moderno-portugues/busca/portugues-brasileiro/agente/. Acesso em: 17 abr. 2020.

MILES, R.; HAMILTON, K. **Learning UML 2.0**. O'Reilly, 2006.

MOODLE USP: **e-Disciplinas**. Disponível em: https://edisciplinas.usp.br/mod/resource/view.php?id=2274122. Acesso em: 08/07/2021. Acesso em: 30 nov. 2022.

PADGHAM, L.; WINIKOFF, M. Prometheus: A practical agent-oriented methodology. In: GIORGINI, B. H.-S. e P. (Ed.). **Agented-Oriented Methodologies**. [S.l.]: IDEA Group Publishing, 2005. p. 107–135.

RUSSEL J. S.; NORVIG, P. **Artificial Intelligence:** a modern approach. 3. ed. 1995. Prentice Hall, Singapore.

SCHREIBER, G. et al. **Knowledge Engineering and Management:** the commonKADS Methodology. [S.l.]: MIT Press, 2000.

SILVA, V. T.; **Uma linguagem de modelagem para sistemas multi-agentes baseada em um framework conceitual para agentes e objetos**. Tese de doutorado. Rio de Janeiro: PUC, Departamento de Informática, 2004.

SOMMERVILLE, I. **Software Engineering**, 9. ed. ISBN-10 137035152, 2011.

UNIVERSIDADE FEDERAL DO CEARÁ. Biblioteca Universitária. **Guia de normalização de trabalhos acadêmicos da Universidade Federal do Ceará**. Fortaleza: Biblioteca Universitária, 2013. Disponível em: https://biblioteca.ufc.br/wp-content/uploads/2019/10/guia-de-citacao-06.10.2019.pdf. Acesso em: 30 nov. 2022.

WOHLIN, C.; RUNESON, P.; HOST, M.; OHLSSON, M.; REGNELL, B.; WESSLÉN, A. **Experimentation in Software Engineering**, Boston, MA: Kluwer Academic Publishers, Springer, 2012.

WOOLDRIDGE, M. **An introduction to Multiagent Systems**. 2. ed. 2002. JOHN WILEY & SONS, LTD

APÊNDICE A – INSTRUMENTO DE COLETA DE DADOS

QUESTIONÁRIO

PERFIL, HISTÓRICO E DADOS DEMOGRÁFICOS

E-mail:

_____

Grau:

( ) Doutor

( ) Aluno de doutorado

( ) Mestre

( ) Aluno de mestrado

( ) Graduado

( ) Aluno de graduação

Cargo na instituição onde trabalha (Alunos preenchem com 'Aluno'):

_____

Instituição onde trabalha:

_____

Quão experiente você é com o desenvolvimento de sistemas multiagentes:

( ) Especialista

( ) Avançado

( ) Intermediário

( ) Iniciante

( ) Sem conhecimento

APRESENTANDO AS FUNDAÇÕES

PSMAR – Um processo para desenvolvimento de sistemas multiagentes com agentes racionais

Vídeo apresentando o processo:
https://www.youtube.com/watch?v=crNipI9OQ04&ab_channel=RendleyArnou
Processo utilizando *BPMN*: https://rendleyarnou.github.io/PSMAR/#list
Ilustração do processo com estudo de caso do Moodle: https://github.com/rendleyarnou/tcc-ii-psmar-cs/tree/main/PSMAR%20MOODLE%20CASE%20STUDY/Moodle%20Docs

QUESTIONÁRIO DE FEEDBACK

1 Você conhece outro processo para apoio ao desenvolvimento de SMA com agentes racionais além do PSMAR?

( ) Sim
( ) Não

Se você conhece outro processo para apoio ao desenvolvimento de sistemas multiagentes com agentes racionais, informe-nos no campo abaixo:

_____

2 É necessária a proposição de um processo para conduzir o desenvolvimento de SMAs com agentes racionais?

( ) Concordo plenamente
( ) Concordo
( ) Não sei
( ) Discordo
( ) Discordo plenamente

3 A especificação textual de requisitos do PSMAR ajuda a estruturar bem o SMA nas fases iniciais do desenvolvimento.

( ) Concordo plenamente
( ) Concordo
( ) Não sei
( ) Discordo
( ) Discordo plenamente

4 O gerenciamento das iterações do PSMAR é relevante para a execução do processo.

( ) Concordo plenamente
( ) Concordo
( ) Não sei
( ) Discordo
( ) Discordo plenamente

5 Regras de mapeamento entre modelagem de requisitos com iStar4RationalAgents e MAS-ML 2.0 são relevantes para o desenvolvimento de SMAs com agentes racionais.
( ) Concordo plenamente
( ) Concordo
( ) Não sei
( ) Discordo
( ) Discordo plenamente

6 Os tutoriais existentes no PSMAR são úteis para facilitar a configuração para modelar com a ferramenta MAS-ML 2.0 e desenvolver com JAMDER.
( ) Concordo plenamente
( ) Concordo
( ) Não sei
( ) Discordo
( ) Discordo plenamente

7 É importante verificar a rastreabilidade dos artefatos gerados.
( ) Concordo plenamente
( ) Concordo
( ) Não sei
( ) Discordo
( ) Discordo plenamente

8 A Planilha e o Relatório de testes são úteis para apoiar os testes de SMAs com agentes racionais.
( ) Concordo plenamente
( ) Concordo

( ) Não sei

( ) Discordo

( ) Discordo plenamente

9 A escolha dos métodos/técnicas e ferramentas para apoio a modelagem de requisitos (iStar4rationalagents), modelagem de projeto (*MAS-ML 2.0*) e *framework* de desenvolvimento (*JAMDER*) foi adequada.

( ) Concordo plenamente

( ) Concordo

( ) Não sei

( ) Discordo

( ) Discordo plenamente

10 Qual valor da escala abaixo representa o nível de contribuição do PSMAR para apoiar o desenvolvimento de SMAs com agentes racionais? (0 não contribui – 5 contribui muito)

( ) 0

( ) 1

( ) 2

( ) 3

( ) 4

( ) 5

Se você acha difícil/muito difícil, qual parte não ficou clara?

_____

11 Qual é o nível de dificuldade para entender o PSMAR?

( ) Muito fácil

( ) Fácil

( ) Média

( ) Difícil

( ) Muito difícil

12 O uso do PSMAR é viável para a criação de SMAs com agentes racionais.

( ) Concordo plenamente

( ) Concordo

( ) Não sei

( ) Discordo

( ) Discordo plenamente

13 Quão adequado é o PSMAR para ser usado na definição de futuros projetos de SMA?

( ) Muito adequado

( ) Adequado

( ) Não sei

( ) Inadequado

( ) Muito inadequado

14 O PSMAR pode ser útil para o desenvolvimento de SMAs pelos desenvolvedores e pesquisadores da área.

( ) Concordo plenamente

( ) Concordo

( ) Não sei

( ) Discordo

( ) Discordo plenamente

15 O PSMAR pode ser útil para criar meus próximos projetos de SMAs.

( ) Concordo plenamente

( ) Concordo

( ) Não sei

( ) Discordo

( ) Discordo plenamente

16 COMO você usará o PSMAR para propor seus próximos projetos de SMAs?

( ) Vou usar o PSMAR como ele foi proposto

( ) Vou usar grandes partes das tarefas e artefatos do PSMAR como foi proposto e fazer algumas adaptações

( ) Vou usar algumas tarefas e artefatos do PSMAR para criar minha instância deste processo

( ) Não vou usar nada do PSMAR

_____

17 A quem o PSMAR pode ser útil?

( ) Desenvolvedores sem experiência no desenvolvimento de SMAs com agentes racionais

( ) Desenvolvedores com experiência no desenvolvimento de SMAs com agentes racionais

( ) Ambos

( ) Discordo

( ) Ninguém

18 Você recomendaria o PSMAR a outros pesquisadores?

( ) Sim

( ) não

( ) Talvez

19 Quais são os pontos fortes do PSMAR?

_____

20 Quais são os pontos fracos do PSMAR?

_____

21 Você tem alterações / melhorias / comentários relacionados ao PSMAR?

_____

22 Você indicaria alguém que atua com desenvolvimento de SMAs para contribuir com esta pesquisa? (Inserir o e-mail no campo abaixo)

_____

APÊNDICE B – DOCUMENTO DE REQUISITOS [MODELO DE DOCUMENTO]

As abordagens baseadas em objetivos concentram-se em porque os sistemas são construídos, fornecendo a motivação e a justificativa para os requisitos de software.

Este documento se destina a especificar os requisitos dos Sistema Multiagente (SMA) para o <**Domínio do Sistema**>. A descrição deste SMA é apresentada em três níveis. O nível 1 é uma descrição de alto nível apresentada de forma textual com intuito de servir de base para o detalhamento nos níveis posteriores. O nível 2 é especificado tomando como base a descrição apresentada no nível 1, neste nível são detalhadas as entidades que compõem o SMA com seus tipos e descrições. Finalmente, o nível 3 é especificado tomando como base o que foi definido nos níveis 1 e 2, detalhando assim, os elementos contidos internamente em cada uma das entidades e seus relacionamentos. O nível 1 é especificado por meio de descrição textual livre, enquanto os níveis 2 e 3 são especificados por meio de tabelas.

NÍVEL 1 - No nível 1 o sistema deve ser descrito textualmente por meio do preenchimento dos campos que em uma abordagem inicial, deve ser coletada uma especificação geral do sistema multiagente. Descreva em alto nível os pontos a seguir para que facilite a compreensão do sistema de modo geral:

1. Qual o domínio do sistema multiagente?
2. O SMA será de simulação ou acoplado a um sistema real?
3. Quais os principais objetivos do sistema e como alcançá-los?
4. No caso de um multiagente que executará acoplado a um sistema real, como será a interação entre os agentes e o sistema?
5. Quais as entidades envolvidas e a relação entre elas?

NÍVEL 2 - No nível 2 as entidades do sistema devem ser identificadas após o preenchimento dos campos do nível 1 na abordagem inicial, deve ser coletada uma descrição específica do sistema multiagente.

Tabela 6 - Ambiente

Ambiente			
ID	**Nome**	**Descrição**	**Tipo**
AMB-001	<Nome do	<Descrição do	<Tipo do ambiente>

	ambiente>	ambiente>

Fonte: Elaborada pelo autor.

Tabela 7 - Organização

Organização		
ID	**Nome**	**Descrição**
ORG-001	<Nome da organização>	<Descrição da organização>

Fonte: Elaborada pelo autor.

Tabela 8 - Agente

Agente		
ID	**Nome**	**Descrição**
AG-001	<Nome do agente>	<Descrição do agente>

Fonte: Elaborada pelo autor.

Tabela 9 - Papéis de agentes

Papéis de agentes			
ID	**Nome**	**Descrição**	**Tipo**
PA-001	<Nome do papel>	<Descrição do papel>	<Tipo do agente que irá receber o papel>

Fonte: Elaborada pelo autor.

Tabela 10 - Relacionamento entre entidades fontes e entidades alvos

Entidade fonte	**Relacionamento**	**Entidade alvo**
<Nome da entidade fonte>	<Tipo de relacionamento entre as entidades>	<Nome da entidade alvo>

Fonte: Elaborada pelo autor.

Nível 3 - No nível 3, as entidades do sistema identificadas após o preenchimento dos campos do nível 2, devem ser coletadas **especificações detalhadas** das entidades do sistema multiagente.

Tabela 11 - Entidade 1

<Nome da entidade>	
<Tipo do elemento interno>	
Nome	**Descrição**
<Nome do elemento interno>	<Descrição do elemento interno>

Fonte: Elaborada pelo autor.

Tabela 12 - Entidade 2

<Nome da entidade>			
<Tipo do elemento interno>		**<Tipo do elemento interno>**	
Nome	**Descrição**	**Nome**	**Descrição**
<Nome do elemento interno>	<Descrição do elemento interno>	<Nome do elemento interno>	<Descrição do elemento interno>

Fonte: Elaborada pelo autor.

Tabela 13 - Entidade 3

<Nome da entidade>		
Entidade fonte	**Relacionamento**	**Entidade alvo**
<Nome da entidade fonte>	<Tipo de relacionamento entre as entidades>	<Nome da entidade alvo>

Fonte: Elaborada pelo autor.

Tabela 14 - Entidade 4

<Nome da entidade>		
Entidade fonte	**Relacionamento**	**Entidade alvo**
<Nome da entidade fonte>	<Tipo de relacionamento>	<Nome da entidade alvo>
<Nome da entidade fonte>	<Tipo de relacionamento>	<Nome da entidade alvo>

Fonte: Elaborada pelo autor.

Modelo de requisitos com *iStar4RationalAgents*

Aqui deverá conter as imagens das modelagens do modelos iStar *SD MODEL* e *SR MODEL* utilizando a ferramenta de modelagem *piStar Tool* com a extensão *iStar4RAtionalAgents*.

APÊNDICE C – PLANILHA DE MAPEAMENTO DE CONSTRUTORES ISTAR4RATIONALAGENTS E MAS-ML 2.0

Tabela 15 - Planilha de mapeamento de construtores iStar4RationalAgents e MAS-ML 2.0

Elementos iStar4RationalAgents	**Elementos *MAS-ML 2.0***
Objetivo	Objetivo
Crença	Crença
Agente reativo simples	Agente reativo simples
Agente reativo baseado em modelo	Agente reativo baseado em modelo
Agente baseado em objetivos	Agente baseado em objetivos
Agente baseado em utilidade	Agente baseado em utilidade
Papel do agente reativo simples	Papel do agente reativo simples
Papel do agente reativo baseado em modelo	Papel do agente reativo baseado em modelo
Papel do agente baseado em objetivos	Papel do agente baseado em objetivos
Papel do agente baseado em utilidade	Papel do agente baseado em utilidade
Função próximo	Função próximo
Função formulação de objetivo	Função formulação de objetivo
Função formulação de problema	Função formulação de problema
Função utilidade	Função utilidade
Ação	Ação
Dever	Dever
Direito	Direito
Causa-efeito	Representado junto das ações dos agentes reativos
Percepção	Percepção
Planejamento	Planejamento
Organização	Organização
Ambiente	Ambiente
Plano	Plano
Agentes, Papéis	Classe
O relacionamento PARTICIPATES IN entre posições, agentes ou funções.	Agregação de classes.

Relacionamento IS A entre posições, agentes ou funções.	Generalização / especialização de classe.
O relacionamento *OCCUPIES* entre um agente e uma posição.	Associação de classe chamada *OCCUPIES*.
O relacionamento *COVERS* entre uma posição e uma função.	Associação de classe chamada *COVERS*.
Relacionamento *PLAYS* entre um agente e uma função.	Associação de classe chamada *PLAY*S.
Tarefas definidas no modelo SD.	Métodos com visibilidade pública.
Tarefas definidas no modelo SR.	Métodos com visibilidade privada.
Recursos definidos no modelo SD.	Classe *IF THIS* dependência tem características de um objeto.
Recursos definidos no modelo SD.	Atributo com visibilidade privada em classe que representa o ator dependente se esta dependência não puder ser caracterizada como um objeto
Recursos (sub-recursos) definidos no modelo SR.	Atributo com visibilidade privada na classe que representa o ator ao qual pertence o sub-recurso (se este sub-recurso não puder ser entendido como um objeto).
Recursos (sub-recursos) definidos no modelo SR.	Uma classe independente, caso contrário.
Metas (leves) no modelo SD.	Atributo com visibilidade pública na classe que representa o dependente.
Metas (leves) no modelo SD.	Atributo com visibilidade pública na classe que representa o ator ao qual pertence o sub objetivo.
Decomposição de tarefas.	Representado por pré e pós-condições (expressas em OCL) da operação UML correspondente.
Goal/Quality	A disjunção dos valores médios implica o valor final.
Objetivo (*Quality*) - Tarefa, Recurso-Tarefa.	A pós-condição da tarefa dos meios implica o valor do fim.
Tarefa-Tarefa	A disjunção da pós-condição dos meios implica nas pós-condições do fim.
Needed-by	Composição
Qualification	Associação
Dependency	*Dependency*

Fonte: Elaborada pelo autor.

APÊNDICE D – CHECKLIST DE RASTREABILIDADE [MODELO DE DOCUMENTO]

Este documento se destina a fazer uma análise de rastreabilidade entre as fases do processo PSMAR para checar a uniformidade do sistema multiagente em questão. Na primeira tabela deverão ser assinaladas as fases do processo que já foram concluídas e que os artefatos estão em conformidade. Na segunda tabela deverão ser assinalados as entidades em conformidade em cada fase do processo.

Tabela 16 - Checklist de rastreabilidade - fases do processo concluídas

Quais subprocessos já foram concluídos?				
[ ]	[ ]	[ ]	[ ]	[ ]
Levantamento e especificação dos requisitos	Detalhar requisitos	Projetar sistema multiagente com *MAS-ML 2.0*	Desenvolver sistema multiagente com *JAMDER*	Testar sistema multiagente

Fonte: Elaborada pelo autor.

Tabela 17 - Checklist de rastreabilidade - entidades do processo concluídas

Entidades	**Requisitos do SMA**		**Projeto do SMA com *MAS-ML 2.0***	**Desenvolvimento do SMA com *JAMDER***	**Teste do SMA**
	Documentos de requisitos	**Modelo de requisitos com iStar4Rational Agents**			
<Entidade-01>	[ ]	[ ]	[ ]	[ ]	[ ]
<Entidade-02>	[ ]	[ ]	[ ]	[ ]	[ ]

Fonte: Elaborada pelo autor.

APÊNDICE E – TUTORIAL DE INSTALAÇÃO DO JADE E JAMDER

Requisitos:

1. Windows 10
2. Eclipse IDE for Java EE Developers

Nota: Ao executar o arquivo de instalação do Eclipse poderão ser apresentadas outras versões do programa, esteja certificado de que irá instalar a versão ***Eclipse IDE for Java Developers***.

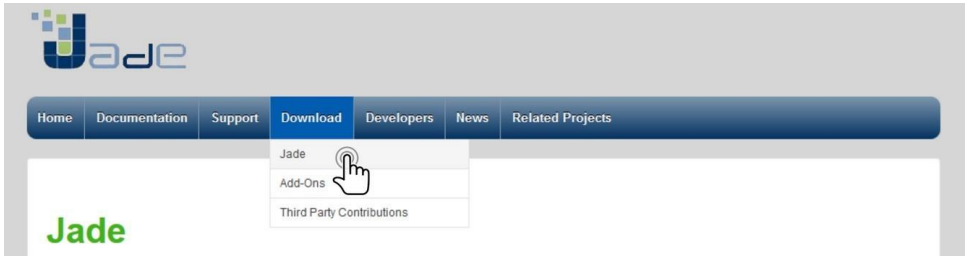
A ferramenta será utilizada para CODIFICAR e não para MODELAR, caso seja instalado outra versão do programa os passos a seguir não irão funcionar.

3. Jade - JAVA Agent Development Framework

(1) Acessar o site oficial do JADE onde o arquivo estará disponível para download: https://jade.tilab.com/

(2) Ao clicar no link acima você será redirecionado para a página inicial, passe o mouse sobre “Download” e clique em “Jade”.

Figura 32 - Página do site https://jade.tilab.com/



Fonte: Elaborada pelo autor.

(3) Após aceitar os termos, clique em:

Figura 33 - Página de download do JADE

JADE	~ File size	Description of the content
jadeAll	17.7 MB	This file contains all JADE, i.e. it is just composed of the 4 files below. If it is too large for downloading, the 4 files below might be downloaded instead.
jadeBin	2.5 MB	This file contains JADE already compiled and ready to be used, i.e. a set of JAVA archive JAR files.
jadeDoc	12.8 MB	This file contains all the JADE documentation included the Administrator's Guide and and the Programmer's Guide. NOTICE THAT all the documentation is also available on-line.
jadeSrc	2.3 MB	This file contains all the JADE source code.
jadeExamples	490 KB	This file contains the source code of the examples and a simple demo. All the examples and demo must be compiled.

Fonte: Elaborada pelo autor.

(4) Após baixar o JADE, você precisará baixar o JAMDER. O arquivo está disponível no link abaixo:

https://github.com/rendleyarnou/tcc-ii-psmar-cs/tree/main/Documentos/6.%20Desenvolver%20sistema%20multiagente%20com%20JAMDER/plugins

(5) Salve na pasta de Downloads ou em uma pasta de sua preferência. Saída:

Figura 34 - Arquivos JAMDER, JADE e Eclipse

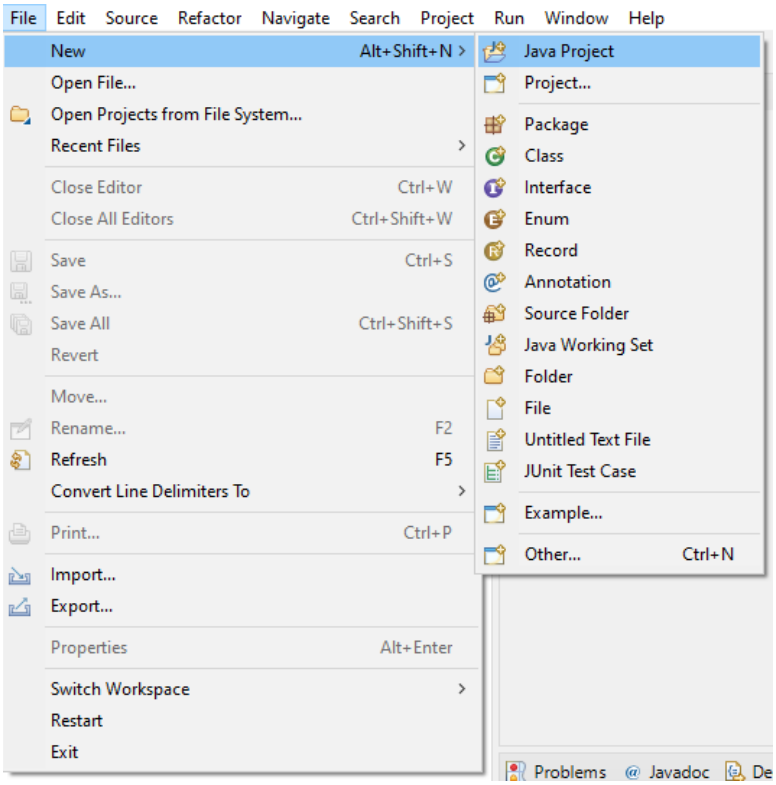
Nome	Tipo	Tamanho
jamder.jar	Arquivo do WinRAR	37 KB
JADE-all-4.5.0.zip	Arquivo ZIP do WinRAR	17.748 KB
eclipse-inst-win64.exe	Aplicativo	53.680 KB

Fonte: Elaborada pelo autor.

Instalação

1. Instalar o Eclipse;
2. Após finalizar o processo de instalação do Eclipse, com o programa em execução crie um novo projeto Java:
    a. File -> New -> Java Project

Figura 35 - Tabela de inicialização de projeto no Eclipse

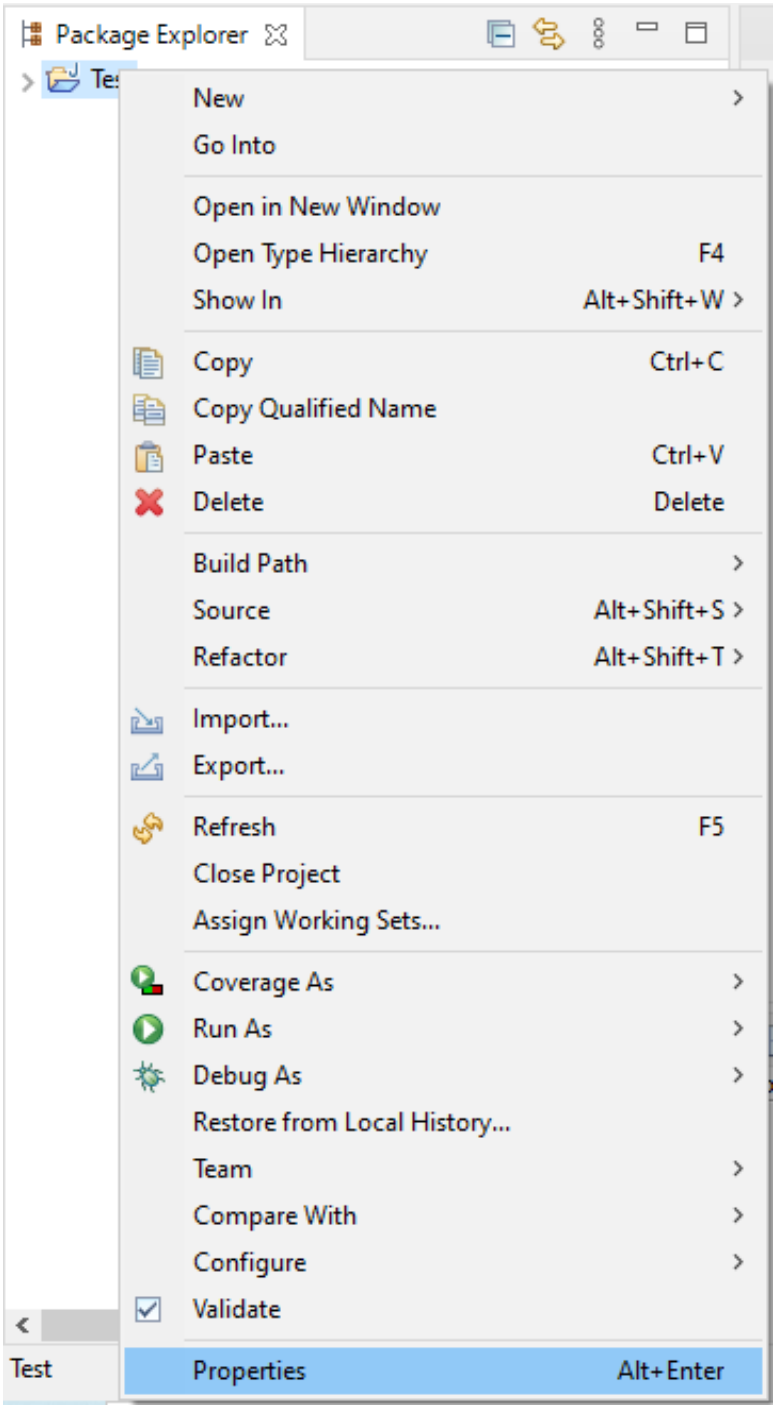


Fonte: Elaborada pelo autor.

3. Após criar o projeto é necessário importar JADE-all-*.zip da seguinte forma:

    a. Clicar com o botão direito do mouse sobre o nome do projeto e em seguida clicar em “Propriedades” (ou properties);

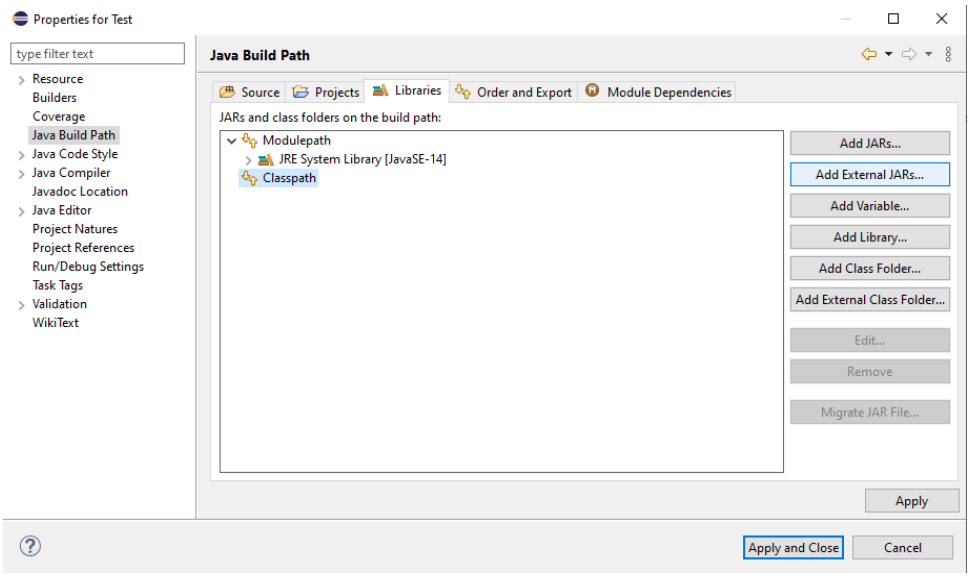
Figura 36 - Selecionar propriedades no Eclipse



Fonte: Elaborada pelo autor.

4. Após o passo anterior, você irá clicar em Java Build Path -> Libraries -> Classpath -> Add External JARs

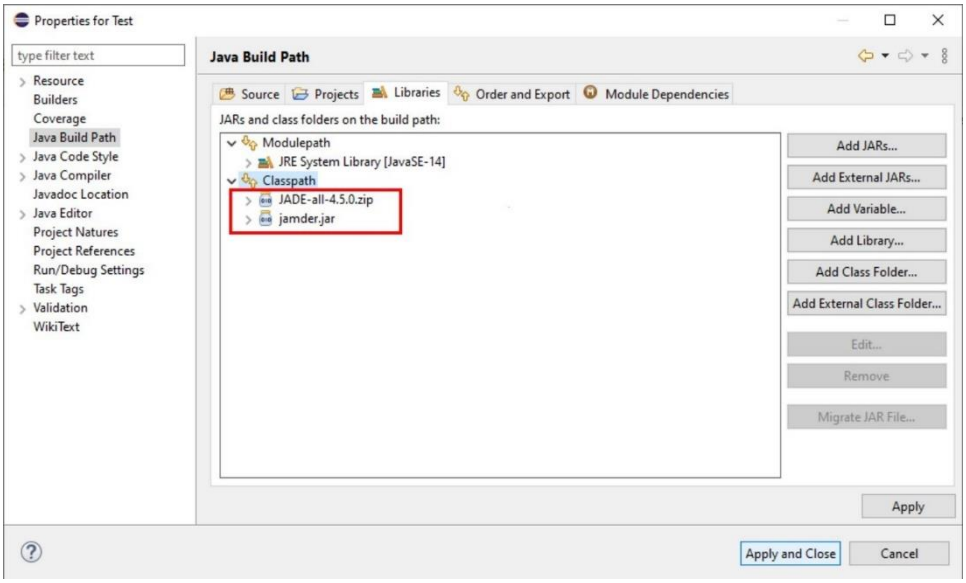
Figura 37 - Propriedades do Eclipse



Fonte: Elaborada pelo autor.

5. Então você deverá selecionaro arquivo JADE-all-*.zip,mesmo procedimento deverá ser feito para adicionar o arquivo jamder.jar.

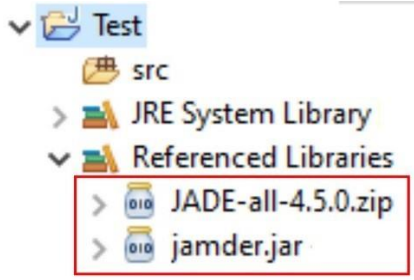
Figura 38 - Extensões nas propriedades do Eclipse



Fonte: Elaborada pelo autor.

6. Após selecionar os dois arquivos você irá clicar em Apply and Close.
7. Ao final da execução deste tutorial a saída deverá ser a seguinte:
    a. JADE e JAMDER adicionados como “Bibliotecas Referenciadas” (Referenced Libraries)

Figura 39 - Extensões JADE e JAMDER



Fonte: Elaborada pelo autor.

APÊNDICE F – PLANILHA DE TESTE [MODELO DE DOCUMENTO]

Tabela 18 - Planilha de teste [MODELO DE DOCUMENTO]

Planilha de teste	
Caso de uso:	<Nome do agente/entidade>
Complexidade:	<Nível de complexidade>
Funcionalidades:	<Funcionalidade do agente/entidade>
Pré-condições:	<Condições para que as funcionalidades sejam satisfeitas>
Observações:	<Observações a serem feitas sobre algo do agente/entidade>
<Funcionalidade 01> **Objetivo da <funcionalidade>: <Descrição do objetivo da funcionalidade>** **Funcionamento da <Funcionalidade>: <Descrição do processo de execução da funcionalidade>** **Pré-requisito para que o objetivo seja alcançado: <Listagem de pré-requisitos que são necessários para que o objetivo seja alcançado>**	
<Cenário da funcionalidade>	<Descrição do cenário>
<1º caso>	<Descrição do 1º caso>
<2º caso>	<Descrição do 2º caso>

Fonte: Elaborada pelo autor.

Tabela 19 - Planilha de teste [MODELO DE DOCUMENTO] - funcionalidade 01

<Funcionalidade 01>	
Dados Válidos	
1) Casos de Teste:	**2) Ideias de Teste:**
1) Sucesso - <Nome do caso de teste>	<Ideia de teste 1> <Ideia de teste 2>
Dados inválidos	
1) Casos de Teste:	**2) Ideias de Teste:**
1) Erro - <Nome do caso de teste>	<Ideia de teste 1> <Ideia de teste 2>

Fonte: Elaborada pelo autor.

APÊNDICE G – RELATÓRIO DE TESTE [MODELO DE DOCUMENTO]

Este documento apresenta o relatório de teste das entidades do sistema multiagente desenvolvido para <finalidade do sma>. Na tabela estão listadas as entidades do sistema multiagente, suas funcionalidades (comportamentos) testados e o resultado de cada um deles (sucesso ou falha).

Tabela 20 - Relatório de teste [MODELO DE DOCUMENTO] - Entidade 01

<Entidade 01>			
Funcionalidade	**Resultado**		**Descrição do erro**
<Fun 01>	Falha [ ]	Sucesso [ ]	
<Fun 02>	Falha [ ]	Sucesso [ ]	
<Fun 03>	Falha [ ]	Sucesso [ ]	
<Fun 04>	Falha [ ]	Sucesso [ ]	
<Fun 05>	Falha [ ]	Sucesso [ ]	

Fonte: Elaborada pelo autor.

Tabela 21 - Relatório de teste [MODELO DE DOCUMENTO] - Entidade 02

<Entidade 02>			
Funcionalidade	**Resultado**		**Descrição do erro**
<Fun 01>	Falha [ ]	Sucesso [ ]	
<Fun 02>	Falha [ ]	Sucesso [ ]	
<Fun 03>	Falha [ ]	Sucesso [ ]	
<Fun 04>	Falha [ ]	Sucesso [ ]	
<Fun 05>	Falha [ ]	Sucesso [ ]	

Fonte: Elaborada pelo autor.

APÊNDICE H – TUTORIAL DE INSTALAÇÃO DO PLUGIN MAS-ML TOOL

Requisitos:

1. Windows 10
2. Eclipse Modeling Tools

   http://www.eclipse.org/downloads/packages/eclipse-modeling-tools/oxygen1

Figura 40 - Ferramenta Eclipse Modeling Tools

Fonte: Elaborada pelo autor.

Nota: Deve ser baixado o Eclipse Modeling Tools, NÃO o Eclipse IDE ou outras versões. Vamos MODELAR e não CODIFICAR, então deve ser usado/baixado o Eclipse Modeling Tools, sendo assim se for utilizado o Eclipse IDE for Java EE Developers, ou qualquer outra que não seja a primeira (Eclipse Modeling Tools), os plugins da MAS-ML NÃO irão funcionar.

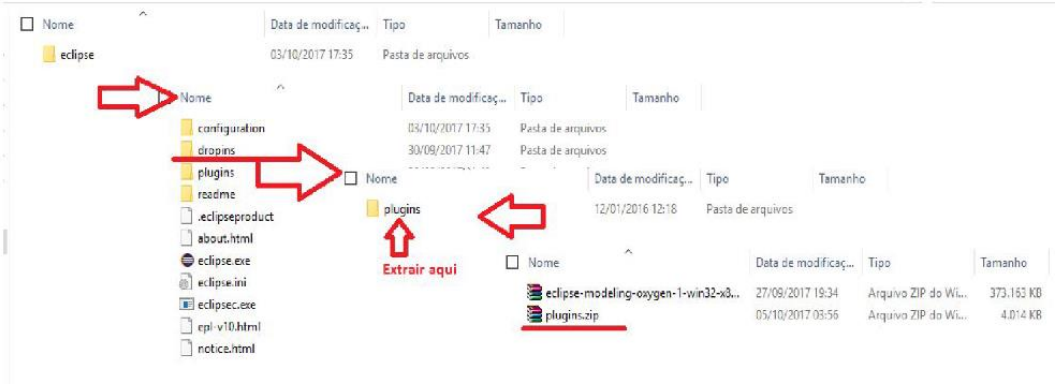
3. Plugins de *MAS-ML 2.0*

   https://github.com/rendleyarnou/tcc-ii-psmar-cs/tree/main/Documentos/5.%20Projetar%20sistema%20multiagente%20com%20MAS-ML%202.0/plugins

Instalação:

1. Extrair o Eclipse
2. Extrair o .zip do Plugins MAS-ML tool baixado
3. Copiar o conteúdo de MAS-ML tool extraído para a pasta **dropins/plugins** do eclipse

Figura 41 - Plugins MAS-ML 2.0

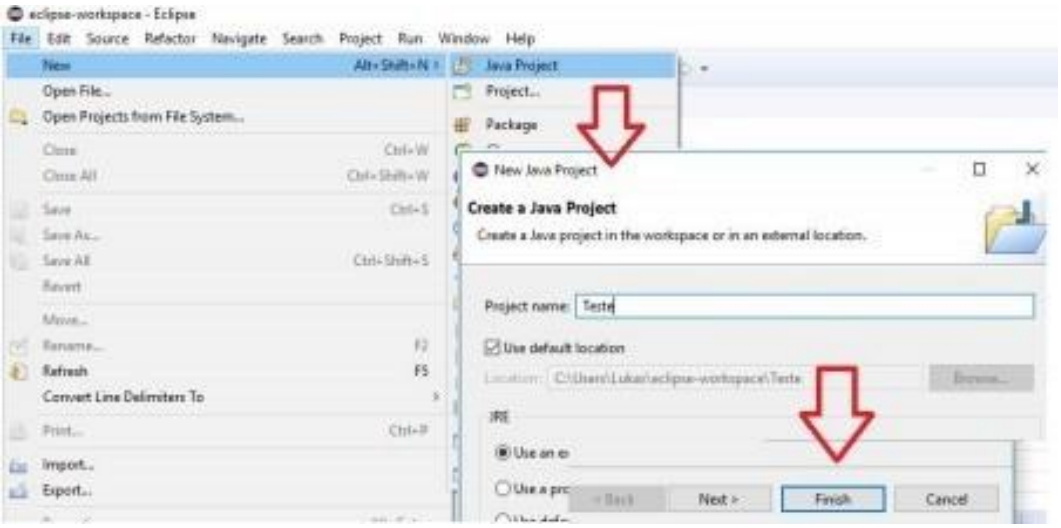


Fonte: Elaborada pelo autor.

Testar funcionamento do plugin:

1. Inicie o Eclipse
2. New>Java Project – nome do projeto (neste exemplo é teste) e finish

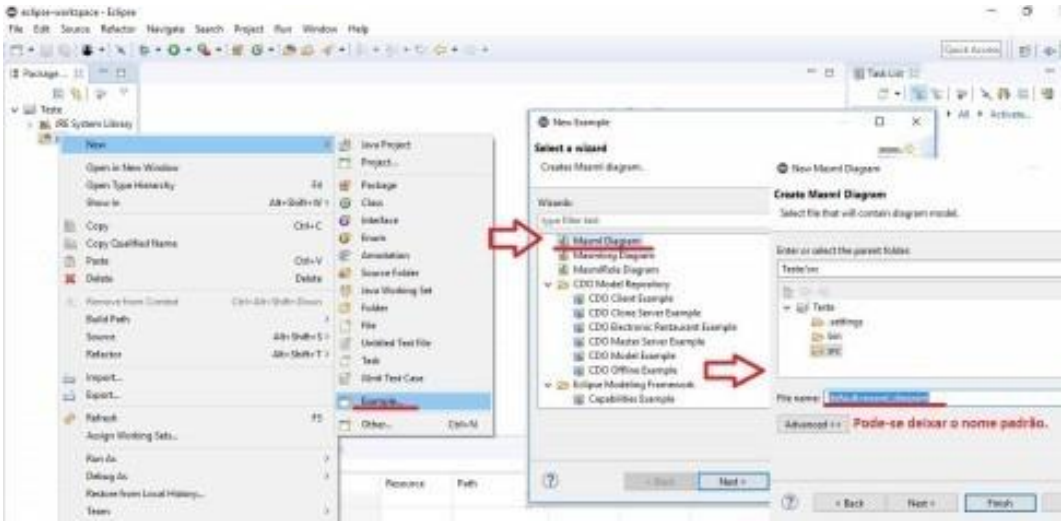
Figura 42 - Tela de criação de novo projeto no Eclipse



Fonte: Elaborada pelo autor.

3. Clique com o botão direito do mouse na pasta src>New>Example>Masml Diagram>Finish

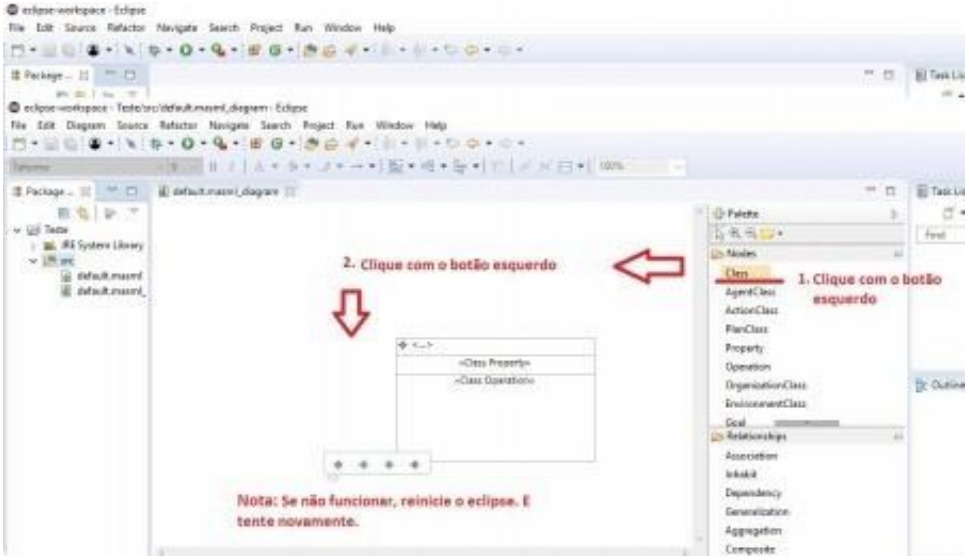
Figura 43 - Finalizando criação de novo projeto no Eclipse



Fonte: Elaborada pelo autor.

4. Agora vamos inserir uma classe

Figura 44 - Inserindo classe no Eclipse



Fonte: Elaborada pelo autor.

APÊNDICE I – DOCUMENTO DE GERENCIAMENTO DE ITERAÇÕES [MODELO DE DOCUMENTO]

Tabela 22 - Documento de gerenciamento de iterações [MODELO DE DOCUMENTO]

Nº da iteração	Funcionalidade	Responsável	Data de início	Data de finalização	STATUS
<Número da iteração>	<Nome da funcionalidade>	<Nome do responsável>	<Data de início>	<Data de finalização>	Feito
<Número da iteração>	<Nome da funcionalidade>	<Nome do responsável>	<Data de início>	<Data de finalização>	A fazer

Fonte: Elaborada pelo autor.