REPMAS: A Requirements Engineering Process for MultiAgent Systems: An Application Example

Iderli Pereira de Souza Filho[a], Gilleanes Thorwald Araujo Guedes[b] and Giovane D'Avila Mendonça[c]

Software Engineering Post-Graduation Program, Pampa Federal University, Av. Tiaraju, 810, Alegrete, Brazil

Keywords: Requirement Engineering Process, Model BDI, Multiagent Systems.

Abstract: Requirements engineering is a crucial phase for the development process of any software, including multi-agent systems. This particular kind of software is composed of agents, autonomous and proactive entities which can collaborate among themselves to achieve a given goal. However, multi-agent systems have some particular requirements that are not normally found in other software, making the requirements engineering general processes and techniques less efficient. Taking this into account, this work presents a specific requirements engineering process for multi-agent systems with emphasis in a consolidated model in the cognitive agents development (BDI Model). This process supports the requirements engineering subareas of elicitation, analysis, specification, and validation, thus, presenting a wide coverage of the requirements engineering area. The proposed process was evaluated through its application in requirements engineering of Heraclito multi-agent system. This assessment allowed us to identify future works and improvement points in our work.

1 INTRODUCTION

Agents technology is a software paradigm that provides agent abstractions for open, distributed, and heterogeneous systems development (Gan et al., 2020). Software agents are autonomous, social, reactive and proactive entities (Hajduk et al., 2019). A multiagent system (MAS) consists of a set of agents that can interact with each other. These systems can be used to solve problems that are difficult or impossible for a monolithic system or for a single agent to solve (Rodríguez Viruel, 2011).

Ferber in (Ferber, 2014) states that developing a MAS is not an easy task. This led to the emergence of AOSE (Agent-Oriented Software Engineering). Slhoub in (Slhoub et al., 2019) states that AOSE aims to support development processes for agent-based systems, imposing a solid practice of software engineering in concepts of artificial intelligence.

AOSE objectives include adapting requirements engineering to the multiagent context. Although there are development processes that address requirements engineering for MAS, we noticed gaps and points for improvement in their approaches which were discovered by means of a primary study, where we carried out a literature systematic review (MendonÇa. et al., 2021). Among these gaps, we considered the most important the need of a process that encompasses the four subareas of requirements engineering defined in SWEBOK (Bourque et al., 2014) and a support to requirements needed to cognitive agents based on the *Belief-Desire-Intention* (BDI) model.

Moreover, according to (Javed et al., 2022) requirements engineering is an essential activity in the software development lifecycle and, according to (Herzig et al., 2017) the concepts of beliefs and goals (desires) play a central role in the conception and implementation of agents. Thus, we realized the need to propose a requirements engineering process for MAS, allowing to represent particular requirements for this kind of system and surpassing the main gaps we found in current processes.

We called this new process REPMAS (Requirements Engineering Process for MultiAgent Systems). We incorporated in REPMAS some existing works for MAS requirements engineering, such as the Homer technique and the MASRML language.

Homer technique allows agent-oriented requirements elicitation and it has the advantage of being easily usable in other approaches (Wilmann and Ster-

[a] https://orcid.org/0000-0001-9694-0977
[b] https://orcid.org/0000-0001-5457-2600
[c] https://orcid.org/0000-0001-9445-6831

Filho, I., Guedes, G. and Mendonça, G.
REPMAS: A Requirements Engineering Process for MultiAgent Systems: An Application Example.
DOI: 10.5220/0011798800003393
In *Proceedings of the 15th International Conference on Agents and Artificial Intelligence (ICAART 2023) - Volume 1*, pages 387-395
ISBN: 978-989-758-623-1; ISSN: 2184-433X

ling, 2005)), while MASRML (Guedes et al., 2020) is a UML-based language that adapts the use-cases diagram in order to support the main concepts used in MAS based on the BDI model. It is important to highlight that our process was validated by means of its application in the second version of the Heraclito system (Galafassi et al., 2019).

This paper is organized as follows, section 2 presents the basic concepts applied in this work. Section 3 compares the proposed process with the other MAS processes that deal somehow with requirements engineering. Our requirements engineering process is detailed in Section 4 and the results of its application in Section 5. Lastly, we present the conclusion.

2 BACKGROUND

2.1 Agents and Multiagent Systems

An agent is a computational process, established in an environment, designed to achieve a purpose by means of autonomous and flexible behavior (Vicari and Gluz, 2007). An agent has autonomy, social ability, reactivity, and pro-activity. Autonomy is the agent's capability to operate without the users' direct intervention. Social ability allows agents to interact among themselves by means of some kind of agent communication language. Reactivity refers to the capability of perceiving the environment and answering timely to the changes that occur in the environment. Finally, pro-activity is the agent capability to have a behavior directed to a goal, taking the initiative without receiving orders (Hajduk et al., 2019).

A multiagent system is composed of several intelligent agents interacting among themselves (Wooldridge, 2009). The development of this kind of system is used in several areas (Liu et al., 2010), due, among other reasons, to its capability to deal with complexity (Boes and Migeon, 2017).

2.2 Belief-Desire-Intention (BDI) Model

The Belief-Desire-Intention model is a software model developed to programming intelligent agents characterized by including beliefs, desires, and intentions in the agent architecture (Bratman et al., 1987). Beliefs represent what an agent believe to be true about the environment, about itself, and about other agents. Desires represent the goals the agent would like to achieve. And intentions represent desires the agent believes he can achieve and take actions to achieve them (Rao and Georgeff, 1995).

2.3 MASRML

MASRML – Multi-Agent Systems Requirements Modeling Language – is a UML-based Domain-Specific Modeling Language (DSML) conceived for the requirements modeling in multiagent system projects (Guedes et al., 2020). This DSML extends the UML metamodel in order that the use-case diagram can be applied in the specific domain of multiagent systems. MASRML provides mechanisms to represent the concepts of agent role, goal (desire), perception, belief, intention, plan, and action, supporting the concepts of the BDI model.

In MASRML, new concepts were created inspired by concepts of UML. The main contribution was the creation of the AgentRoleActor and InternalUseCase metaclasses to represent agent roles and the functionalities assigned to them, stereotyped in goals, perceptions, actions, and plans. Some associations were also created in order to associate perceptions, actions, and plans to the goals that an agent role wishes to achieve, thus establishing the conditions for a goal to become an intention and which plan will be executed in this case, as well as the possible external actions performed during the execution of a plan.

Furthermore, MASRML internal use-cases documentation provides a clear view of the structure of cognitive agents, allowing, for example, to determine what is needed for a given goal to become an intention or to establish the steps to be performed when executing a perception or a plan.

REPMAS uses MASRML notation throughout its execution, producing partial MASRML use-case diagrams for each agent role and a final MASRML use-case diagram representing all agent role actors and all the internal use cases assigned to each one of them.

2.4 Requirement Engineering

In SWEBOK (Bourque et al., 2014) it is stated that the requirements engineering process cover four main subareas: (I) Requirements Elicitation; (II) Requirements Analysis; (III) Requirements Specification; and (IV) Requirements Validation.

Requirements elicitation aims to understand the problem that the software must solve using techniques such as interviews, scenarios, prototyping, observations, user histories, among others. Requirements analysis aims to detect and solve conflicts among the requirements, to classify the requirements, to derive new software requirements, and to establish how the functionalities must interact with its organizational and operational environment.

Requirements specification, by its turn, produces

requirements documents that can be systematically reviewed, evaluated, and approved. Finally, requirements validation evaluates requirements documents to ensure that the requirements are understandable, consistent, and complete.

2.5 The Homer Technique

Homer (*Human Oriented Method for Eliciting Requirements*), proposed by (Wilmann and Sterling, 2005), is a technique for eliciting requirements proper to MAS. When applying this technique, the interviewer asks the Stakeholders questions through organizational metaphors, as making the stakeholders think in the software as a company that is hiring new employees. This style of elicitation aims to more easily discover the agent roles and its goals within the system, besides the system boundaries, business rules, and relationships between the agents.

It is important to highlight that Homer assumes that the use of multiagent systems is valid and that the customer desires an agent-oriented solution. However, Homer does not perform a feasibility test to determine whether a multiagent system is suitable for solving the problem or not.

2.6 The Heraclito System

Heraclito is a multiagent system for the interactive and dialectical learning with focus on the natural deduction teaching of propositional logic (Galafassi et al., 2019). This Intelligent Tutor System is based in a Multiagent System and identifies the individual knowledge of each student in the context of Natural Deduction in Propositional Logic.

Heraclito assists students in solving various types of Logic exercises and provides the LOGOS Electronic Notebook to create and edit formulas, truth tables and proofs of Propositional Logic (Galafassi et al., 2019). We applied REPMAS in the requirements engineering of Heraclito second version.

3 RELATED WORKS

With the objective to understand the state-of-art of the requirement engineering processes for multiagent systems, mainly in what concerns to the coverage of the requirements engineering subareas defined in SWEBOK (Bourque et al., 2014) and its support to the BDI model (Rao et al., 1995), we performed a literature systematic review (MendonÇa. et al., 2021).

The systematic review analysed a total of 43 studies and did not find any methodology that supports the use of BDI model and cover the four requirements engineering subareas defined in SWEBOK. Moreover, we perceived a strong lack, in a general way, in the areas of requirements elicitation and validation.

This review was the basis for our proposal of a MAS requirements engineering process, since we search for acting in the gaps found in the current processes and contributing to the area as a whole.

The support to the BDI model was mentioned in 8 processes, however, performing a deeper analysis, we realized that only the PRACTIONIST process (Morreale et al., 2006) supports the use of this model during requirements engineering. Nevertheless, this process only contemplates the requirements analysis and specification subareas, it does not support the elicitation and validation phases.

4 REPMAS - A REQUIREMENTS ENGINEERING PROCESS FOR MULTIAGENT SYSTEMS

We identified several processes that cover requirements engineering for multiagent systems through our literature systematic review (MendonÇa. et al., 2021). The focus of our study was to observe the coverage of requirements engineering of these processes and their support for the BDI model in order to identify gaps in both requirements engineering and BDI support.

From this systematic review, we identified the need to create a requirements engineering process for multiagent systems. Thus, we developed the REPMAS process with the aim of covering all the requirements engineering subareas defined in SWEBOK (Bourque et al., 2014) adapted specifically for the context of multiagent systems.

It is important to highlight that our process also aims to adapt requirements specific for MAS, such as agent roles, perceptions, goals, and actions. In this way, we intend to provide a greater support to requirements engineering for cognitive agents.

Another relevant point to be highlighted is that, although this process is limited to Requirements Engineering, our research group is working on techniques and notations that can help to expand this process to other areas of Software Engineering, such as the Software Design subarea.

4.1 General Process Description

We believe it is important to highlight the choice of using the Homer technique and the MASRML language. We decided to use them in REPMAS because

both were created to fill gaps in requirements engineering for MAS. According to Willmann (Wilmann and Sterling, 2005), Homer technique was conceived due to the little attention AOSE devotes to requirements models and notations focused on elicitation. And, according to Guedes (Guedes et al., 2020). MASRML was created to solve the lack of mechanisms to model functional requirements for multiagent systems.

Thus, as can be seen in Figure 1, REPMAS starts in the requirements elicitation subprocess (detailed in Figure 2), that generates the preliminary requirements document. After eliciting the requirements, the analyst must identify the roles that agents can play. Agent roles can be discovered by verifying which functionalities can be described in the preliminary scenarios and grouping these functionalities by speciality.

For each agent role, the subprocess of agents internal scenarios identification (detailed in Figure 3) is executed. In this subprocess are generated a partial MASRML use-case diagrams (UCDs) for each agent goal, resulting in the phase where the final UCD is produced. After producing the final UCD, the documents created must be validated.

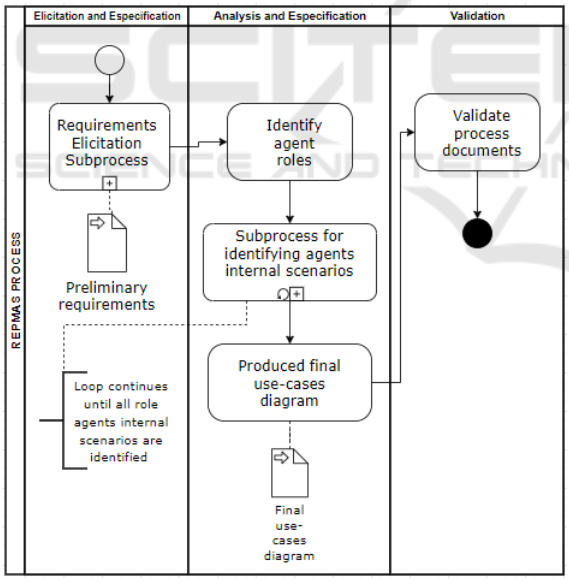

Figure 1: General Process.

4.2 Requirements Elicitation Subprocess

Analysing the processes of requirements engineering for multiagent systems, we verified that Homer technique (Wilmann and Sterling, 2005), stood out among the analysed methodologies. This was the only technique that presented a way to elicit requirements focused directly with the stakeholder. Moreover, Homer was made specifically to be integrated into other development process, that is the reason why we decided to integrate this technique in our process.

Another important point is that none methodology presented an appropriate approach for the BDI model, we verified that Tropos methodology (Mylopoulos et al., 2013) supports BDI, however although Tropos states to have an elicitation phase, it does not present ways for eliciting requirements. Tropos elicitation phase is not described, this appears to be a kind of free phase where it is allowed to its users to apply the elicitation technique they want.

Thus, we understand that Homer technique is the most suitable for collecting specific requirements for multiagent systems and we integrate this technique into our process. The requirements elicitation subprocess can be seen in Figure 2.

The application of these questions intends to identify generic characteristics of the environment to extract the preliminar scenarios that will serve as a basis to the agents internal scenarios identification and to the diagrams production.

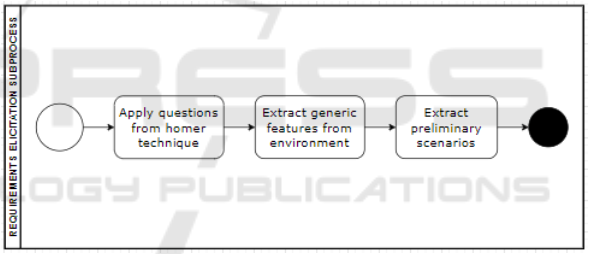

Figure 2: Subprocess of Requirements Elicitation.

4.3 Subprocess of Agent Internal Scenarios Identification

The subprocess of agent internal scenarios identification (Figure 3), intends to extract agents specific characteristics as well as documenting internal use cases and producing partial use-cases diagrams.

For each agent role defined previously, the analyst must identify the initial beliefs associated with this role and identify a goal for this agent role. With the identification of this goal, the analist must determine the beliefs needed for the goal to become an intention, identify perceptions needed for the identified goal, determine which beliefs can be affected by the perceptions identified and the possible external actions associated with the goal.

The analyst must verify whether for this goal is necessary a plan. It is possible that a goal does not have plans associated (in situations in which the agent

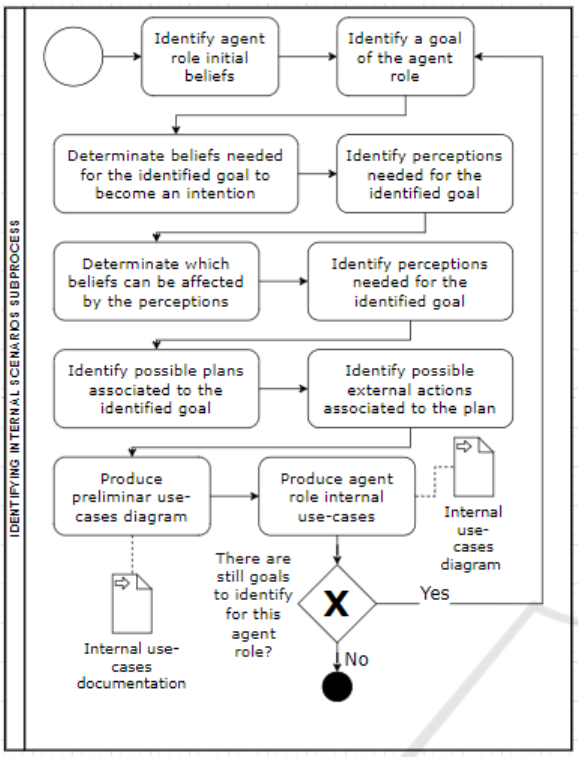

Figure 3: Subprocess of Agent Internal Scenarios.

reactions are simple and demand little reasoning), however it is also possible for a goal to have more than a plan, it will depend on the goal complexity. If it is realized that a plan is necessary, then it is needed to identify the steps to accomplish that plan and the external actions associated with the plan.

The subprocess is finished with the production of a partial MASRML use-case diagram and the documentation of the scenarios relative to the use cases that compose it. This subprocess presents how is made the requirements specfication with the application of MASRML. This DSML allows to model BDI agents, demonstrating its cogntive features.

It is important to highlight that the documentation of MASRML scenarios presents a description of the system functioning and agents' characteristics, in addition to what is presented in the diagrams, such as the agents' beliefs.

5 PROCESS EXECUTION

This section has as its objective to present the execution of the proposed process in the developing of the requirements engineering for the second version of the Heraclito System.

5.1 Applying the Requirements Elicitation

To verify the functioning and applicability of Homer, we applied this technique in interviews with the stakeholders of the Heraclito project. We hope that the requirements engineering performed in this work will contribute to the development of the second version of Heraclito.

We performed three interviews with Heraclito system stakeholders. It is important to highlight that the conflicts found during the interviews were solved by means of new interviews with the stakeholders.

The performed interviews allowed to extract the positions in the organizational metaphor proposed and its tasks in the enterprise. We also got to extract the dependencies relations between the positions and its limitations. Heraclito system does not have a position hierarchy, however we believe that it would be possible to delimit a hierarchy with these questions for other systems.

By means of applying Homer technique, we got to extract the agent roles (defined by positions in the organizational metaphor) and the goals of these roles. We also verified whether these goals depend on other goals to be executed. For each goal, we also extracted tasks for the goal to be achieved. Besides these characteristics, the Homer technique would allow us to extract the hierarchy levels of each agent role, however, we did not identify hierarchy levels in the agent roles in the Heraclito system.

With the content elicited we were able to create a preliminary description of Heraclito scenarios. These scenarios allowed us to identify agent roles and to develop the use-case models representing the system internal scenarios.

However, we would like to highlight that we were not able to elicit the functionalities accessed by the external agents (users), because Homer technique does not cover the users actions. In this case, we asked questions not included in Homer, with the objective to understand which functionalities could be accessed by the system users.

5.2 Identification Agent Roles

Applying Homer methodology questions allowed us to acquire a description of the functions needed in the organization.

When verifying the answers, we noticed that all the interviewees presented three base-functions for Heraclito. However, the interviewee 2 showed answers very disconnected from the others. Analysing his answers we realized that several parts of the sys-

tem were missing. On the other hand, the interviewees 1 and 3, provided very similar answers, mostly changing only the nomenclature.

Regarding the answers of the interviewee 2, we made the decision to use those that were compatible with those of the others interviewee and to validate in other interviews those answers that were disconnected with the answers of the others interviewee.

From the description of these functions, we delimited the system agents. Next, analysing the answers, not only to the questions, but also of the tasks described, we got to identify similarities in the agents described, defining the following agent roles:

- **Technical Specialist:** It is responsible for all that needs technical knowledge. This agent role prepares the student content, corrects tests and questions, and follows the student in each action in solving a question;
- **Mediator:** It is the single responsible for intermediating the system with the student. It follows the student development, presenting feedback and providing clues. This agent role also knows the student profile;
- **Bayesian Analyst:** This agent role has knowledge about the historical of the student questions and tests. Based on this historical, the Bayesian Analyst prepares a Bayesian Network containing several statistics of the student results.

We also highlighted that the agents contained in the Heraclito System are cognitive agents, i. e., these agents present a mechanism of decision taking, advanced interactions, and have goals strongly established.

5.3 Production of the Agents Internal Scenarios

Aiming to demonstrate the operation of MASRML, together with the process execution, in this section we will present a scenario analysed from the Heraclito System.

The partial use-case diagram (Figure 4), presents the necessary steps for the Mediator Agent Role to perform the ”Inform question to the Student” internal functionaly. The agent playing this role perceives the solicitation of a new question and the goal ”Inform question to the Student” became an intention. This way, the plan ”Prepare to apply the question” is triggered, ending with the presenting of the question to the student. The description of the scenario steps needed to perform this Goal can be viewed in the Tables 1, 2, 3, and 4.

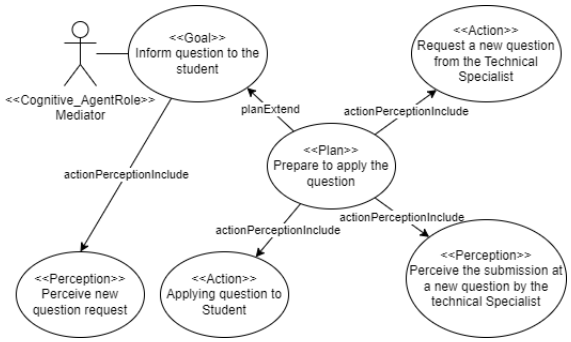

Figure 4: Partial Use-Case Diagram - Inform a Question to the Student.

Table 1: Goal - Inform Question to the Student.

Internal Use Case Name	Inform Question to the Student.
Stereotype	Goal
AgentRoleActor	Mediator
Abstract	This internal use case describes the steps followed by the agent that plays the role of Mediator to inform new study questions to the student.
Initial Beliefs	Negative solicitation of a new question
Perceptions	Probing of new solicitations of questions
Main Scenario	
Actions to AgentRoleActor	
1. Execute the internal use case ”Perceive new question request”	
Alternative Scenario - Solicitation of a New Question Perceived	
AgentRoleActor Actions	
1. Change belief of Solicitation of a New Question to positive	
2. Make the goal an intention.	
3. Execute internal use case ”Prepare question application”	

Table 2: Perception - Perceive a new question request.

Internal Use Case Name	Perceive a new question request
Stereotype	Perception
AgentRoleActor	Mediator
Abstract	This internal use case describes the probing of the environment to verify if a student asked for a new study question.
Pre-conditions	The Goal Internal Use Case ”Inform question to the student” must be in execution.
Initial Beliefs	There was no request for a new question.
Main Scenario	
AgentRoleActor Actions	
1. Probe the environment to verify the occurrence of a new request of question.	
Alternative Scenario - True Perception	
AgentRoleActor Actions	
1. Starts to believe that a student requested a new question.	

5.4 Requirements Validation

In the requirements validation we verified the acceptance of the proposed scenarios. This review validation was applied after the first version of the scenarios specification and served as a method to identify errors in the scenario description, together with a validation of the agent's goals.

This validation was made by presenting the sce-

Table 3: Perception - Perceive the submission at a new question by the technical Specialist.

Internal Use Case Name	Perceive the submission at a new question by the Technical Specialist
Stereotype	Perception
AgentRoleActor	Mediator
Abstract	This internal use case describes the probing of the environment to verify if the Technical Specialist sent a new question.
Pre-conditions	The Plan Internal Use Case "Prepare question application" must be in execution.
Initial Beliefs	Technical Specialist did not send a new question.
Main Scenario	
AgentRoleActor Actions	
1. Probe the environment to verify if a new question from the student was sent by the Technical Specialist.	
Alternative Scenario - New question perceived.	
AgentRoleActor Actions	
1. Starts to believe that the Technical Specialis sent a new question.	
2. Receives a new question from the Technical Specialist.	

Table 4: Plan - Prepare to apply the question.

Internal Use Case Name	Prepare to apply the question
Stereotype	Plan
AgentRoleActor	Mediator
Abstract	This internal use case describes the steps followed by the agent that plays the Mediator role to request a question to the Technical Specialist to allow its application.
Main Scenario	
AgentRoleActor Actions	
1. Request a new question to the Technical Specialist	
2. Execute internal use case "Perceive the submission of a new question by the Technical Specialist"	
Alternative Scenario - Sending of new question from the student perceived	
AgentRoleActor Actions	
1. Apply student question.	

narios to the stakeholders, asking if they were correct and asking for suggestions. If the stakeholder found an error or if there was something missing in the scenario description, a comment suggesting alterations in the scenario should be made.

With this validation we identified several errors presented in the specification first version. Among the errors found there was the lack of depth in some goals and points not covered in the interviews.

Even containing errors, the validation gave us a confirmation that the elicitation and analysis were satisfactory. The requirements elicited and analysed were mostly correct and the stakeholders were satisfied with the documentation produced.

5.5 Improvements Foressen to the Process

Applying REPMAS in a real environment, allowed us to extract some information that can contribute with future works.

The technique Homer has several weaknesses, most of them regarding its coverage or difficulty to extract some particular requirements for MAS. However, we considered Homer organizational metaphor very useful for extracting requirements for MAS.

Currently, in our research group, we are developing a technique for requirements elicitation based on the organizational metaphor present in Homer technique. We intend to include this technique in the initial phase of this requirements engineering process, in order to make it easier to obtain scenarios that will be analyzed and specified with help of MASRML.

Another consideration we can quote is that the agents' beliefs are not expressed in the models. MASRML use-case models, in spite of supporting particular MAS requirements, do not provide a way to express the agents' beliefs, only in the use cases documentation. So it is necessary to perform a study to verify if there are ways of belief modeling or if it is needed to propose a method and/or notation for it. This study is being performed in parallel with this work and we are verifying the possibility to model beliefs through conceptual models produced by extended UML class diagrams to represent agents' beliefs, besides other structural information as goals, perceptions, and plans representations.

Finally, from a wider perspective, we considered that our work contributed to the requirements engineering of the Heraclito project in a satisfactory way, eliciting and analysing a wide range of specific requirements for MAS, specially when dealing with BDI agents.

6 CONCLUSIONS AND FUTURE WORKS

In this study we proposed REPMAS - a requirements engineering process for multiagent systems. The proposed process encompassed the four requirements engineering subareas inspired in the SWEBOK (Bourque et al., 2014): elicitation, analysis, specification, and validation. With this process we tried to give support to the software engineer when performing a requirements engineering for this kind of system.

The elicitation stage of this process aims to discover specific requirements for MAS by means of a direct contact with the Stakeholder, eliciting software agents and its functions, by means of interview, performing an organization metaphor with the hiring of new employees for an enterprise. This elicitation was performed by means of the Homer methodology (Wilmann and Sterling, 2005). We have also made

an evaluation of this methodology execution, where there were found several weak points that can serve as a base for future works.

The analysis was proposed to be performed by means of scenario identification, produced by means of MASRML (Guedes, 2012), a language that extends UML use-case diagram for the context of multiagent systems. This analysis aims to identify specific requirements for MAS focusing on the support of the BDI model. Finally, the validation phase of this process aims to validate the artifacts specified in the elicitation and analysis phases.

To evaluate the proposed process, we applied it in the second version of the Heraclito systems (Galafassi et al., 2019) - a multiagent system that aims to aid in logical teaching. The requirements engineering presented in this process can help in the development of the second version of this system.

In the process execution, we performed interviews with Heraclito Stakeholders, with the objective of eliciting the requirements and to identify preliminar scenarios for the system. These preliminar scenarios served as the base to the elaboration of the preliminar use case diagrams, in which we identified the goals, perceptions, plans, and actions of the agent roles of Heraclito System. We validated these scenarios by means of a complete reading with the stakeholders.

Based on this study, we identified opportunities for future work. One of them is related to a possible extension of the Homer methodology, to provide a bigger support to particular requirements of MAS, such as actions, perceptions, and plans. Another opportunity for future work is the need of a notation that expresses the agents' beliefs and how they can change. Moreover, we are currently extending our process in order to adapt other phases of software engineering for the multi-agent systems development.

REFERENCES

Boes, J. and Migeon, F. (2017). Self-organizing multi-agent systems for the control of complex systems. *Journal of Systems and Software*, 134:12–28.

Bourque, P., Fairley, R. E., et al. (2014). *Guide to the software engineering body of knowledge (SWEBOK (R)): Version 3.0*. IEEE Computer Society Press, Washington, DC, USA.

Bratman, M. et al. (1987). *Intention, plans, and practical reason*, volume 10. Harvard University Press Cambridge, MA, Chicago, USA.

Ferber (2014). *Handbook on agent-oriented design processes, prefácio*. Springer, USA.

Galafassi, F. F. P., Galafassi, C., Vicari, R. M., and Gluz, J. C. (2019). Heráclito: Intelligent tutoring system for logic. In *International Conference on Practical Applications of Agents and Multi-Agent Systems*, pages 251–254. Springer, Cham.

Gan, K., Anthony, P., On, C., and Hamdan, A. (2020). *Enforcing Social Semantic in FIPA-ACL Using SPIN*, pages 3–13. Springer, Singapore.

Guedes, G., Souza F., I., Gaedicke, L., Mendonça, G. D., Vicari, R., and Brusius, C. (2020). Masrml - a domain-specific modeling language for multi-agent systems requirements. *International Journal of Software Engineering & Applications (IJSEA)*, 11(5):21.

Guedes, G. T. A. (2012). *Um metamodelo UML para a modelagem de requisitos em projetos de sistemas multiagentes*. PhD thesis, Universidade Federal do Rio Grande do Sul (UFRGS).

Hajduk, M., Sukop, M., and Haun, M. (2019). *Cognitive Multi-agent Systems*. Springer, USA.

Herzig, A., Lorini, E., Perrussel, L., and Xiao, Z. (2017). Bdi logics for bdi architectures: old problems, new perspectives. *KI-Künstliche Intelligenz*, 31(1):73–83.

Javed, S., Alam, K. A., Ajmal, S., and Iqbal, U. (2022). Requirements engineering education: A systematic literature review. In *Proceedings of International Conference on Information Technology and Applications*, pages 469–480. Springer.

Liu, G., Zheng, L., and Zhu, T. (2010). Cooperative planning process modeling on daily dispatching plan in railway bureau based on multi-agent. In *2010 International Conference on Intelligent System Design and Engineering Application*, volume 2, pages 616–621.

MendonÇa., G., Filho., I., and Guedes., G. (2021). A systematic review about requirements engineering processes for multi-agent systems. In *Proceedings of the 13th International Conference on Agents and Artificial Intelligence - Volume 1: ICAART,*, pages 69–79, USA. INSTICC, SciTePress.

Morreale, V., Bonura, S., Francaviglia, G., Centineo, F., Cossentino, M., and Gaglio, S. (2006). Goal-oriented development of bdi agents: the practionist approach. *2006 IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, 1(10221975):66–72.

Mylopoulos, J., Castro, J., and Kolp, M. (2013). *The Evolution of Tropos*, pages 281–287. Springer Berlin Heidelberg, Berlin, Heidelberg.

Rao, A. S. and Georgeff, M. P. (1995). Bdi agents: From theory to practice. In *in proceedings of the first International Conference on Multi-Agent Systems (ICMAS-95*, pages 312–319, San Francisco, CA, USA. ICMAS.

Rao, A. S., Georgeff, M. P., et al. (1995). Bdi agents: from theory to practice. In *ICMAS*, volume 95, pages 312–319.

Rodríguez Viruel, M. L. (2011). *Requirements Modeling for Multi-Agent Systems*, chapter 1, pages 3–22. InTech, Asunción, Paraguay.

Slhoub, K., Carvalho, M., and Nembhard, F. (2019). Evaluation and comparison of agent-oriented methodologies: A software engineering viewpoint. In *2019 IEEE International Systems Conference (SysCon)*, pages 1–8, Orlando, FL, USA. IEEE.

Vicari, R. M. and Gluz, J. C. (2007). An intelligent tutoring system (its) view on aose. *International Journal of Agent-Oriented Software Engineering*, 1(3-4):295–333.

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

Wooldridge, M. (2009). *An introduction to multiagent systems*. John Wiley & Sons, USA.