A Unified Interaction Model with Agent, Organization, and Environment

Maicon R. Zatelli[1], **Jomi F. Hübner**[1]

[1] Department of Automation and Systems Engineering
Federal University of Santa Catarina (UFSC) – Florianópolis, SC – Brazil

{maicon,jomi}@das.ufsc.br

Abstract. *The interaction in multi-agent systems (MASs) is already a quite studied theme, but there are still some open issues. Beyond the interaction between agents, it is possible to see the interaction between agents and environment, or between agents and organization. These other interaction kinds allow us to conceive several situations that are not limited to speech acts. Since most of current work do not consider these extended view of interaction, in this paper we propose an interaction model that considers the environment, organization, and agents. We also show how to integrate this model into a multi-agent platform.*

1. Introduction

The AEIO approach (*Agent*, *Environment*, *Interaction*, *Organization*) [Demazeau 1995, Hammer et al. 2006] has conceived a multi-agent system (MAS) as composed of four basic components: agents, environment, interaction, and organization. In this approach, the MAS developers should choose the model of agents, environment, interaction, and organization which will be instantiated to solve a particular problem. Therefore, an MAS is not based only on the existence of agents, but there are other elements equally important. The developer should be able to see each of these parts clearly and separately.

Nowadays, it already exists many works about the agent, organization, and environment. There are tools to specify, develop, and execute each one. For example, an MAS developer is able to build the environment by means of CArtAgO [Ricci et al. 2006, Ricci et al. 2009], the organization by means of Moise [Hübner et al. 2002], AGR [Ferber et al. 2003], ISLANDER [Esteva et al. 2004], and so forth, and finally, the agents by means of JADE [Braubach et al. 2005], Jason [Bordini et al. 2007], 2APL [Dastani and Meyer 2008], and so on. There are also tools to link these components to work together, such as JaCaMo [Boissier et al. 2011]. However, none of the current tools provide us features to specify and execute the interaction considering the existence of the three other components, that is, to allow the interaction between agents, between agents and environment, and between agents and organization.

In this paper, our aim is to propose a *unified* and *coherent* model for interaction regarding the organization, environment, and agent. The main idea is to institutionalize how the agents must interact with the different elements in an MAS. Unified means a single approach for the interaction integration with the other three components. It is unnecessary to use other combined approaches to get an interaction component integrated with the other components. Coherent means that the concepts used in the interaction are the same as those used in the environment or organization. That is, the role concept existing in the interaction is the same role concept in the organization. The goal concept in the

interaction is the same goal concept in the organization. The same equivalence between concepts occurs in the environment, where there are concepts like artifacts, observable events, operations, and so forth. In the proposed approach we have mapped some necessary concepts existing in each component onto the concepts used into the interaction model.

This paper is organized as follows. In section 2, we present some related work and the current state-of-art relative to interaction. In the following, in section 3, we propose a model to integrate the interaction component into an MAS, and how to specify a protocol using our approach. Finally, before conclusion, we show some results and discussion.

2. Interaction in MAS

In this section, we present the interaction problematic and some related work. We start with the works focused on interaction between agents, followed by those that consider the interaction with the environment, and in the following, the works that regard the interaction with the organization. Finishing this section, we mention some works that have already introduced the interaction problematic to consider the integration with the three other components.

2.1. Interaction - Agent

Related with the interaction between agents, some works are already pointing the drawbacks of keeping the interaction specification inside the agents code [Doi et al. 2005, Miller and McBurney 2007, Singh 2011]:

- The disperse code affects the maintainability of the application;
- When it is necessary to modify some protocol, the code of every agent involved in the interaction has to be modified;
- The protocols could not be composed in run-time in order to permit more complex interactions.

Some approaches defend the separation between the interaction code and the agents code as a solution for these problems. It is unnecessary to keep the interaction control inside the agents code [Miller and McBurney 2007, Miller and McBurney 2008, Oliva et al. 2008, Kubera et al. 2008, Singh 2011]. The separation of the two issues simplifies the development of applications, leading to a modular approach [Giacomo et al. 2003]. Consequently, protocols could be used to compose more complex protocols [Paurobally and Cunningham 2002, Huget and Vitteau 2003, Cabac et al. 2003, Paurobally and Cunningham 2003, Desai et al. 2005, Desai and Singh 2007]. In [Huget and Vitteau 2003, Paurobally and Cunningham 2003, Singh 2011], it is presented other advantages of a modular approach such as the specification of reusable protocols, the improvements in the validation process, and the capacity to share protocols between agents at run-time.

2.2. Interaction - Environment

Keeping the agents and interaction code separated is not the only issue. One of the main limitation in most of works is to regard the interaction only by means

of message exchange, not considering the agent interaction with the environment [Bel-Enguix and Jimenez-Lopez 2007, Baldoni et al. 2010, Baldoni et al. 2011]. Some examples that justify this kind of interaction are presented in [Baldoni et al. 2010, Baldoni et al. 2011]. One of these examples refers to the election in human world. When the people have to do some election, they do not say the candidate name. They use their hands to interact with the electronic ballot box or even simply raise them without saying any word. On the one hand, the electronic ballot box is responsible for computing the votes and notify the winner. On the other hand, by raising their hands, people also may discover the winner of some election only by counting the upper hands. In both cases, the interaction occurs by actions and perceptions in the environment and not by speech acts.

The environment is already considered in different ways to work with interaction [Platon et al. 2005, Keil and Goldin 2005, Saunier and Balbo 2009]. In [Platon et al. 2005] and [Saunier and Balbo 2009], it is presented a model that allows some different kinds of interaction, called overhearing, or eavesdropping. In this interaction type, the agent may intercept messages of others by using the environment. The environment is a way to send and receive messages. In [Keil and Goldin 2005], the aim is to provide an environment as a way to permit indirect interaction. Their focus is on interactions type like stigmergy, which are interactions used by several natural systems such as amoebae and ants.

2.3. Interaction - Organization

The organization is another important component to consider during the interactions. For instance, the GAIA methodology [Wooldridge et al. 2000] has already defined a role as a composition of four main attributes: responsibilities, permissions, activities, and protocols. The protocols are responsible for specifying the interaction between the agents that are playing the organizational roles.

Some works follow the GAIA methodology and provide ways to regulate the interaction by using an organizational model ([Esteva et al. 2004, Ferber et al. 2003, Dignum et al. 2004, Boissier et al. 2010, Hübner et al. 2010]). The interaction protocols are specified inside an organization model, by creating a dialogical dimension. In this case, they use several organization concepts, like goals and roles. Each of these concepts are strongly connected with the interaction concepts.

2.4. Integration with the Three Components

A set of works already have an initial integration between interaction and the other three components [Silva et al. 2004, DeLoach and Valenzuela 2006, Oliva et al. 2008, Kubera et al. 2008, Baldoni et al. 2010, Baldoni et al. 2011], but their aim is different than ours. For example, their interaction specification is conceived to be handled by humans during the MAS design and do not allow the agents to read it (or eventually to change it) at run-time. In [Oliva et al. 2008], it is only considered, in the protocol specification, the interaction between agents. The environment is a mediator between agents interaction. Another existing limitation in this work is that it does not have an integration with an organization model. A role, for example, while existing inside a protocol, may not exist in the organization. As a consequence, the specification may lack coherence since different role conceptualization exist in different components. In [Kubera et al. 2008], the environment is considered by another perspective: the agents could recognize other agents

by the concept of neighborhood. The agents are only able to communicate with others depending on how far they are from each other. As the first work, it does not consider the actions or perceptions performed by the agents in the environment, and the organization component is rather simple (only role names are considered).

The MERCURIO framework [Baldoni et al. 2010, Baldoni et al. 2011], a very similar work with ours, focus on integration of the interaction model regarding agents and environment. The environment conception considers the actions performed by the agents and the perceptions that the agents may sense. The limitation of MERCURIO is related to the organizational component. The roles in the interaction are not strongly connected with the roles existing in the organization. The existence of the other organizational concepts is not considered either. Indeed, the main aim of MERCURIO is to deploy the interaction with the environment.

In contrast to these three previous works, MAS-ML [Silva et al. 2004] and O-MaSE [DeLoach and Valenzuela 2006] are two methodologies that consider the interaction integration with the three other components. However, both approaches are conceived for the specification phase, not regarding the implementation and execution phases. In this case, there are still some shortcomings. For example, these methodologies do not provide a feature to generate the interaction code.

As presented in this section we can notice the lack of proposals that regard the interaction integration with the three other components. Although some authors are concerned with the interaction between agents and some of the other components, none of them integrate the interaction with all three components by proposing a single approach. MAS-ML and O-MaSE gave a first step in order to provide an approach such that, however, their main objective is to provide tools to specify the MAS by using the software engineering concepts, such as diagrams, documents, and so on. Therefore, none of the current approaches regard the interaction in the implementation and execution phases.

3. Proposed Model

The proposed model links the three existing components and the interaction component. The Figure 1 shows the model diagram. In the next subsections we explain how the integration with each component has been done.

3.1. Interaction Protocol

The main concept of our interaction component is the *interaction protocol*[1]. Each protocol is composed of one or more states (Figure 1), and each state is composed of zero or more transitions. A transition is the link between two states and it is responsible for pointing the next state. A transition is fired by means of message exchange, action performance, or some observable event in the environment. Finally, in each transition has the entity responsible for firing the transition, and the entity that will undergo some action, perceive some event, or receive some message. An entity may be an agent playing some organizational role or an artifact in the environment.

A protocol example, which is possible to represent by using our approach, is depicted in Figure 2. In this protocol we have two agents (Ana and Bob), where Ana wants

[1]Due lack of space we will only explain a simplified version of a protocol.

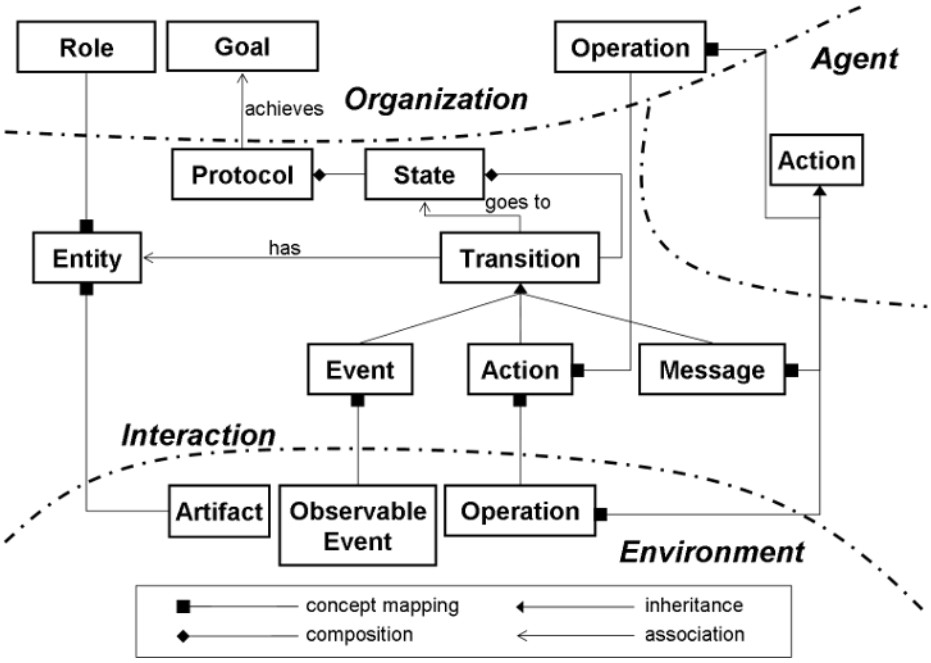

Figure 1. Proposed model with AEIO approach components.

to buy some product `X`. This is a common scenario in the human world when people go to some store to buy products. In the protocol, Ana is an agent playing the role `client` whilst Bob is an agent playing the role `seller`. The organization is represented by means of a generic store wherein the agents may change their roles. The environment is represented by means of two artifacts called sales system and electronic payment terminal (EPT). The sales system is an artifact that provides operations by means of menus, forms, and buttons, and shows some observable properties by means of its display. This artifact is useful to provide for the store manager a control of the sales that had been done, and help the cashiers to calculate the total of the sales. The EPT is an artifact that allows some agent to provide its password with security and also to zip its credit card. In addition, this artifact interacts with the sales system, informing the payment success. For the sake of simplicity, we assume that Ana and Bob already are playing their respective roles, and the environment and organization are also configured and ready to be used.

In the begin, Ana sends a message to Bob asking the price of the product `X`. Bob replies with the price `P` and then Ana decides to buy some amount `A` of the product `X`. At this time, Bob has to adopt the role `cashier`, because he will process the sale and receive the payment. It is important to note that the process to adopt some role is done by means of an interaction between the agent Bob with the organization by using the action `adoptRole(cashier)`.

The next step is the agent Bob starting to ring up the sale. Bob executes an action `newSale` in the sales system informing the product `X`, its price, and its amount. After it, the sales system shows the total `T`. Ana informs Bob that she wants to pay for her purchase by credit card. Bob sets the payment form to credit card in the sales system. Again, it is important to note the interactions between the agent Bob with the environment by means of the sales system artifact. The actions `newSale(X, P, A)` and `paymentForm(card)` are done in the environment. It is also possible to identify the

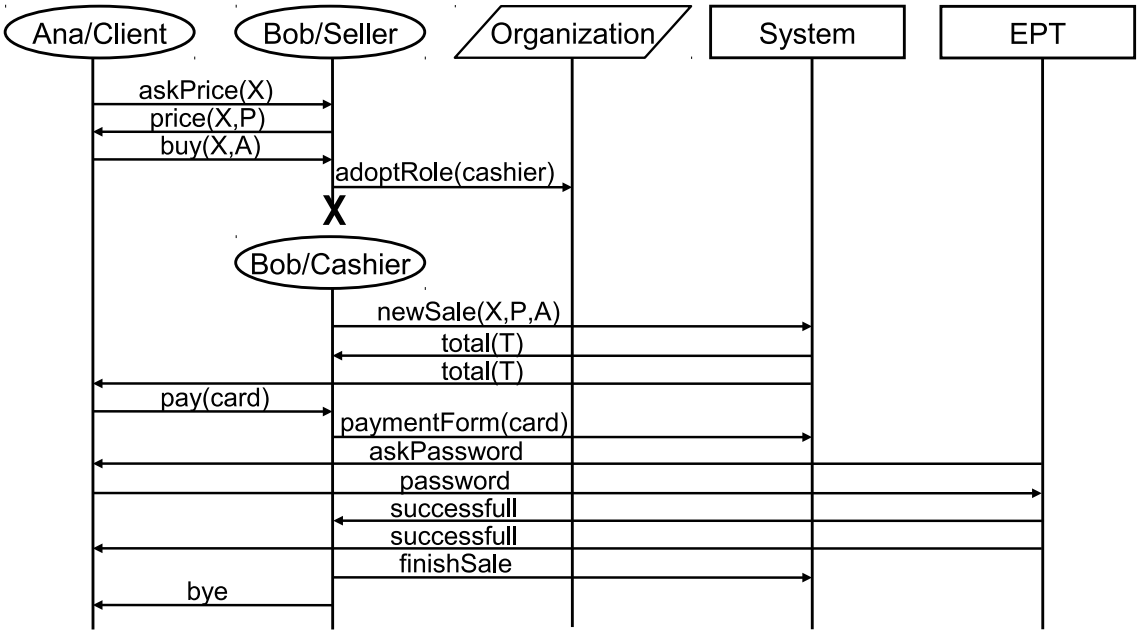

Figure 2. Example of a protocol.

observable event `total(T)`, which occurs when the environment changes some of its properties.

After this step, Ana zips her credit card through the EPT and the EPT displays a message for Ana asking her password. Then, Ana informs her password to the EPT. It is important to note that her password is not sent to Bob. Bob can not discover the Ana's password. In order to guarantee it, Ana only interacts with the EPT. Other EPT function is to check her password and then emits a signal confirming the payment. Both, Ana and Bob perceive this event, and then Bob may finish the sale by executing the action `finishSale`. This action will decrease the product stock and get the sales system ready for a next sale. In the last step, Bob sends a bye message to Ana, meaning the end of the sale. This message ends the protocol execution.

The relevant points in this protocol is the participation of not only agents, but also the environment and the organization. With our approach it is possible represent such protocol and also execute it.

3.2. Linking the Agent

The actions performed by the agents in an MAS link the agents and the interaction component (Figure 1). An agent action can be of three different kinds. The first are the actions performed in the organization. Some examples of this organizational action are adopt or leave some role, commit to some mission or goal, and achieve some goal. The second kind of action is the actions performed in the environment. The environment may have many different elements (or artifacts), allowing the agents to execute a variety of operations depending what the elements in the environment provide. For example, suppose the existence of some calculator artifact. The agent may execute some operations such as addition, subtraction, multiplication, division, square, and so on. Finally, the third kind of action concept is the message exchange. The message exchange comprehends the communicative acts performed by the agents in order to talk with other agents.

In our model (Figure 1), we map the concept of agent action onto our interaction model. The actions performed in the organization are mapped onto operations in an organization. The actions performed in the environment are mapped onto operations in an environment. In the end, the message exchange is mapped onto messages in the interaction component.

3.3. Linking the Organization

The goals, roles, and the operations link the organization and the interaction component (Figure 1). A protocol is used to achieve a goal in the organization component, whilst the roles are used to allow an agent to play the protocol. The operations are performed by the agent and are mapped onto actions in the interaction component. We also have linked the roles of the organization model into the interaction model. There is no role in the interaction model that does not exist in the organization.

This coherence is fundamental to allow the agents to use the interaction in the best way. For example, the agent should not send messages or search hardly for a protocol participant. It may figure out the other participants fast, only by looking at the organization specification. As a role defines the capabilities of each agent, looking at the organization is the correct way to figure out their partners. Therefore, it is important to keep the coherence between the models to preserve this advantage.

3.4. Linking the Environment

The actions performed by the agents in the environment, the signals or observable events emitted by the environment, and the artifacts link the environment and the interaction component (Figure 1). Both action and observable event are used during the protocol execution. An alarm is an example of an observable event that the environment may emit. The alarm event, for example, may notify the agents in respect of the next step of a protocol. It warns the agents to leave some place that is in fire or to put out the fire. The interaction model needs also to handle this event.

Similar to the operations performed in the organization, the operations in the environment are mapped onto actions in the interaction component. The observable events are also mapped onto our model. They are mapped onto events in the interaction model. Finally, the artifact is an entity that may be used to fire some transition by means of observable events or by undergoing some action performed by an agent. In both cases, this concept is mapped onto entity in the interaction component. Again, it is important to keep the coherence between the artifact concept and the entity concept.

3.5. Getting the Interaction Information

One of the most important issues that we have to concern in the integration of our model into an MAS platform is how to get the information exchanged during the MAS execution. Some works like [Ancona et al. 2012] receive the information by means of an agent mediator, which has the task to intercept the messages and to handle the interaction protocol. This mediator informs the agent to proceed or to withdraw with the message. In [Oliva et al. 2008, Baldoni et al. 2011], the messages are intercepted in low level. There is a layer that allows the agents to send all messages, but only the correct messages are used by a protocol to change its state. The advantage to intercept messages in low level is

that the agents could not cheat the interaction. Indeed, this latter option is more appropriate. The former is easy for an agent to send a correct message to the agent mediator, but send a wrong message to continue the protocol. That is, it is easy to cheat the mediator agent. Due the agent autonomy, it is also possible that the agent mediator is malicious.

The Figure 3 shows our proposal of interception model, which put the interaction to work together with the other components. It shows messages, actions, and events being intercepted during their occurrences. There is no way for the agents to change the protocol state sending a wrong message or performing a wrong action. For example, in the protocol presented in the Figure 2, Bob must send a message to Ana with the price of the product X. When Bob sends this message it is intercepted and handled by the interaction before this message arrive to Ana. After it, the interaction sends to Ana the next protocol step. If the message is wrong, the protocol does not change its state and Ana does not receive a new protocol step.

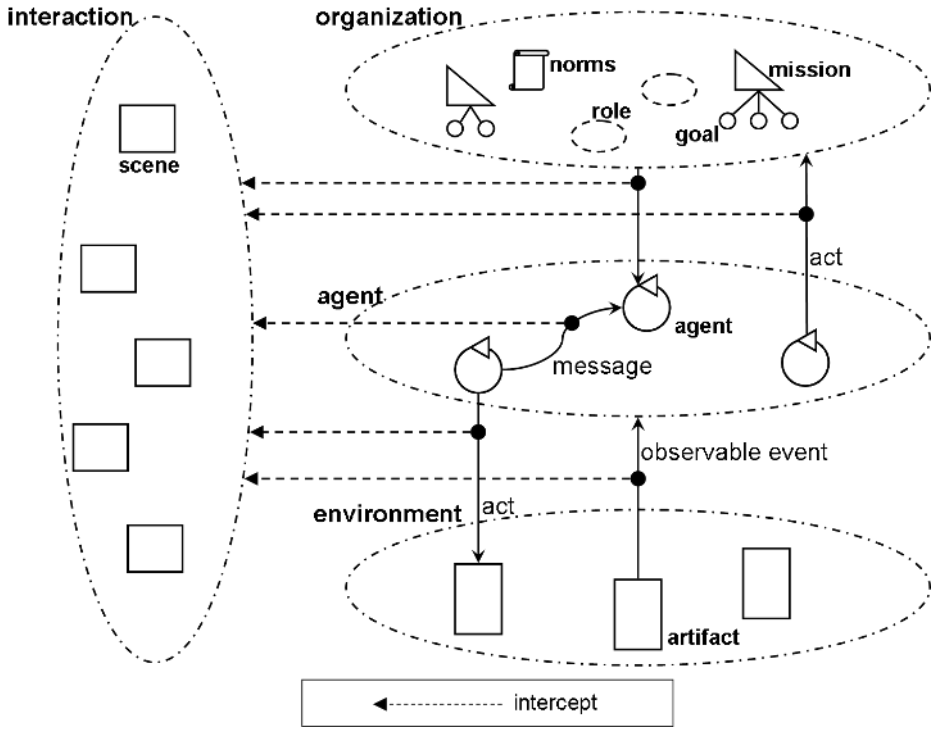

Figure 3. Interception model.

In order to validate our approach we have integrated[2] it into JaCaMo platform [Boissier et al. 2011]. JaCaMo is a project that has the objective to permit the developer to split each one of the MAS components. Currently, it regards the agent, environment, and organization components, but the interaction component is not integrated adequately.

4. Results and Discussion

Our main contribution is the proposed model and its initial specification. This model allows to get the interaction information between each MAS component and use them to guide the agents to achieve the organizational goals. We also provide a single approach

[2]Due the lack of space we will not show the implementation details in this paper.

to integrate the interaction with the other components. The result is a unified proposal to handle the interaction component. Furthermore, by keeping the same concepts in the interaction and in the other components we achieve coherence. The coherence is important because it lets us see the relations between the interaction and the other components separately and clearly.

The proposal helps the agents to accomplish their goals. When an agent adopts some role in the organization, the agent receives the list of goals that it needs to achieve. In order to achieve them, the agent may look at the interaction component for a protocol that may be used to do that. It is only possible thanks to the coherence, because the goals in the interaction are the same in the organization. It is useful because, sometimes, the agents may not know how to proceed to achieve their goals, then the protocol helps the agents in this situation by means of a well-defined sequence of steps. In addition, when the agent accomplish the protocol, the interaction satisfies the goal. It is possible thanks to the link between goal and protocol. It is unnecessary for the agent to change the goal state.

Other coherence contribution is during the protocol instantiation. At this time, the agents have to choose the participants. Again, as the roles in the interaction are the same in the organization, it is sufficient for the agents to read the organization model and to look for the agents that are playing some role. A similar situation may occur with the environment by means of the artifacts. In order to fill out the artifact participants it is sufficient for the agents to look for the artifact instances in the environment that are of the required type.

The MAS security is also improved. Every message may have the participant checked with the organization model to forbid messages from agents playing wrong roles. For example, if the next message has to be sent by a role `seller`, the only agents that could send this message are agents playing the role `seller` in the organization. If other agent tries to send the message, pretending to be a `seller`, it is easy to detect and block it. Other security issue is about the correct action or message in terms of content. The agents could not cheat the protocol, because the messages and actions may be intercepted. If some agent tries to execute some action or send some message that are unexpected it is possible to intercept it and ignore it. Both issues allow us to keep the protocol in the same state. Finally, as the agents are playing some organizational role and they are obliged to follow the organizational norms it is possible to create mechanisms to punish the agents when they intend to act maliciously.

As our proposal considers the interaction with the environment by means of actions or perceptions, it is possible to represent how the agents have to proceed to interact with the artifacts. An advantage of the actions specification that the agents have to do, is that the agents do not need to learn all functions of an artifact. They only need to learn the actions that they must use. But, without a protocol to specify it, the agents have to test or learn all functions in order to use the correct function in a specific case. For example, suppose a case that some agent must use some artifact to handle some situation. If the agent does not receive the actions that it has to do, it needs to read the artifact manual in order to search the correct operation to use. As in our model the necessary actions are pointed by some protocol, the agent already know what function it has to use, since this action is strongly connected with some operation in the artifact.

Finally, the model also allows the use of open MAS where the agents may be heterogeneous. The separation of the interaction component of the other MAS parts endow the system with this capacity. The agents could join the MAS, plays some role and reads the interaction protocols in run-time. The agent is able to learn new protocols and to communicate with other ones or even with the environment by means of artifacts. In addition, if the agent knows how to follow a protocol, we can change the protocol specification without change the agent code. Even in the case of open and heterogeneous MAS, a global behavior can be defined for the overall system.

5. Conclusions and Future Works

In this paper we proposed a new approach to use the interaction strongly connected with the environment and organization model and not only with the agents. In order to achieve this aim, we have developed an interaction component that makes a link with the environment, organization, and agents. The necessary concepts of each component are mapped onto the interaction. We have also shown how to specify protocols by using the new approach and how to integrate it into an MAS.

In the future, we intend to evaluate the approach by means of some parameters such as robustness, performance, and scalability. We have also the idea to test other ways to represent the protocol or even to execute it. There are technologies like Petri Nets, state charts, normative languages, logic languages, and so on, which already are used by other approaches.

References

Ancona, D., Drossopoulou, S., and Mascardi, V. (2012). Automatic generation of self-monitoring mass from multiparty global session types in jason. In *Proc. of DALT*.

Baldoni, M., Baroglio, C., Bergenti, F., Boccalatte, A., Marengo, E., Martelli, M., Mascardi, V., Padovani, L., Patti, V., Ricci, A., Rossi, G., and Santi, A. (2010). Mercurio: An interaction-oriented framework for designing, verifying and programming multi-agent systems. In *Proc. of MALLOW*, pages 134–149.

Baldoni, M., Baroglio, C., Bergenti, F., Marengo, E., Mascardi, V., Patti, V., Ricci, A., and Santi, A. (2011). An interaction-oriented agent framework for open environments. In *Proc. of AI*IA*, pages 68–79, Berlin, Heidelberg. Springer-Verlag.

Bel-Enguix, G. and Jimenez-Lopez, M. D. (2007). Agent-environment interaction in a multi-agent system: a formal model. In *Proc. of GECCO*, pages 2607–2612, New York, NY, USA. ACM.

Boissier, O., Balbo, F., and Badeig, F. (2010). Controlling multi-party interaction within normative multi-agent organizations. In *Proc. of MALLOW*, pages 17–32.

Boissier, O., Bordini, R. H., Hübner, J. F., Ricci, A., and Santi, A. (2011). Multi-agent oriented programming with jacamo. *Science of Computer Programming*.

Bordini, R. H., Hübner, J. F., and Wooldridge, M. (2007). *Programming multi-agent systems in AgentSpeak using Jason*. Wiley, Liverpool.

Braubach, L., Pokahr, E., and Lamersdorf, W. (2005). Jadex: A bdi agent system combining middleware and reasoning. In *Ch. of Software Agent-Based Applications, Platforms and Development Kits*, pages 143–168. Birkhaeuser.

Cabac, L., Moldt, D., and Rolke, H. (2003). A proposal for structuring petri net-based agent interaction protocols. In *Proc. of ICATPN*, pages 102–120, Berlin, Heidelberg. Springer-Verlag.

Dastani, M. and Meyer, J. C. (2008). A practical agent programming language. In *Proc. of ProMAS*, pages 107–123, Berlin, Heidelberg. Springer-Verlag.

DeLoach, S. A. and Valenzuela, J. L. (2006). An agent-environment interaction model. In *Proc. of AOSE*, pages 1–18, Berlin, Heidelberg. Springer-Verlag.

Demazeau, Y. (1995). From interactions to collective behaviour in agent-based systems. In *Proc. of EuroCogSci, Saint-Malo*, pages 117–132.

Desai, N., Mallya, A. U., Chopra, A. K., and Singh, M. P. (2005). Owl-p: A methodology for business process development. In *Proc. of AOIS*, pages 79–94.

Desai, N. and Singh, M. P. (2007). A modular action description language for protocol composition. In *Proc. of AAAI*, pages 962–967. AAAI Press.

Dignum, V., Vázquez-salceda, J., and Dignum, F. (2004). Omni: Introducing social structure, norms and ontologies into agent organizations. In *Proc. of PROMAS*, pages 181–198. Springer.

Doi, T., Tahara, Y., and Honiden, S. (2005). Iom/t: an interaction description language for multi-agent systems. In *Proc. of AAMAS*, pages 778–785, New York, NY, USA. ACM.

Esteva, M., Rosell, B., Rodriguez-Aguilar, J. A., and Arcos, J. L. (2004). Ameli: An agent-based middleware for electronic institutions. In *Proc. of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 1*, Proc. of AAMAS, pages 236–243, Washington, DC, USA. IEEE Computer Society.

Ferber, J., Gutknecht, O., and Michel, F. (2003). From agents to organizations: An organizational view of multi-agent systems. In *Proc. of AOSE*, pages 214–230. Springer Verlag.

Giacomo, C., Ferrari, L., and Leonardi, L. (2003). Brain: a framework for flexible role-based interactions. In *Proc. of CoopIS*, pages 145–161. Springer.

Hammer, F., Derakhshan, A., Demazeau, Y., and Lund, H. H. (2006). A multi-agent approach to social human behaviour in children's play. In *Proc. of IAT, Washington*, pages 403–406.

Huget, M. and Vitteau, B. (2003). Modularity in interaction protocols. In *Workshop on Agent Communication Languages'03*, pages 291–309.

Hübner, A., Dimuro, G. P., Costa, A. C. R., and Mattos, V. L. D. (2010). A dialogic dimension for the moise+ organization model. In *Proc. of MALLOW*, pages 21–26.

Hübner, J. F., Sichman, J. S., and Boissier, O. (2002). A model for the structural, functional, and deontic specification of organizations in multiagent systems. In *Proc. of SBIA*, pages 118–128, London, UK, UK. Springer-Verlag.

Keil, D. and Goldin, D. Q. (2005). Indirect interaction in environments for multi-agent systems. In *Proc. of E4MAS*, pages 68–87.

Kubera, Y., Mathieu, P., and Picault, S. (2008). Interaction-oriented agent simulations: From theory to implementation. In *Proc. of ECAI*, pages 383–387, Patras, Greece. IOS Press.

Miller, T. and McBurney, P. (2007). Using constraints and process algebra for specification of first-class agent interaction protocols. In *Proc. of ESAW*, pages 245–264, Berlin, Heidelberg. Springer-Verlag.

Miller, T. and McBurney, P. (2008). On illegal composition of first-class agent interaction protocols. In *Proc. of ACSE*, pages 127–136, Darlinghurst, Australia, Australia. Australian Computer Society, Inc.

Oliva, E., Viroli, M., Omicini, A., and Mcburney, P. (2008). Argumentation in multi-agent systems. chapter Argumentation and Artifact for Dialogue Support, pages 107–121. Springer-Verlag, Berlin, Heidelberg.

Paurobally, S. and Cunningham, J. (2002). Achieving common interaction protocols in open agent environments. In *Proc. of AAMAS*.

Paurobally, S. and Cunningham, J. (2003). Developing agent interaction protocols using graphical and logical methodologies. In *Proc. of PROMAS*, pages 149–168. Springer.

Platon, E., Sabouret, N., and Honiden, S. (2005). Overhearing and direct interactions: point of view of an active environment, a preliminary study. In *Proc. of E4MAS*, pages 121–138. Springer Verlag.

Ricci, A., Piunti, M., Viroli, M., and Omicini, A. (2009). Environment programming in cartago. In *Multi-Agent Programming II: Languages, Platforms and Applications, Multiagent Systems, Artificial Societies, and Simulated Organizations*. Springer.

Ricci, A., Viroli, M., and Omicini, A. (2006). CArtAgO: An infrastructure for engineering computational environments in MAS. In Weyns, D., Parunak, H. V. D., and Michel, F., editors, *Proc. of E4MAS)*, pages 102–119, AAMAS 2006, Hakodate, Japan.

Saunier, J. and Balbo, F. (2009). Regulated multi-party communications and context awareness through the environment. In *MAGS*, pages 75–91.

Silva, V. T., Choren, R., and de Lucena, C. J. P. (2004). A uml based approach for modeling and implementing multi-agent systems. In *Proc. of AAMAS*, pages 914–921, Washington, DC, USA. IEEE Computer Society.

Singh, M. P. (2011). Information-driven interaction-oriented programming: Bspl, the blindingly simple protocol language. In *Proc. of AAMAS*, pages 491–598.

Wooldridge, M., Jennings, N. R., and Kinny, D. (2000). The gaia methodology for agent-oriented analysis and design. *Autonomous Agents and Multi-Agent Systems*, pages 285–312.