Extending JaCaMo for organizational interoperability

Tomas M. Vitorello, Fabio T. Muramatsu, Anarosa A. F. Brandão

Computer Engineering and Digital Systems Department
Escola Politécnica – Universidade de São Paulo (USP)
CEP – 05508-900 – São Paulo – SP – Brazil

{tomas.vitorello, fabio.muramatsu, anarosa.brandao}@usp.br

Abstract. *Traditionally organization-centered multiagent systems (OC-MAS) are strongly dependent on their own organizational infrastructure. This dependence is one of the main concerns when trying to achieve interoperability among them. In this paper, we investigate the possibility of addressing the issue by interpreting the organization as a set of norms imposed upon the agents. This work is based on the JaCaMo infrastructure, which already implements the Moise+ organization model through this approach. Our aim is to extend this platform to support other organizational models, effectively allowing agents to interoperate among organizations described according to different organization models.*

1. Introduction

The development of open Multiagent Systems (MAS) faces a natural difficulty when the aim is to achieve a collective goal through the interaction of self-interested, autonomous agents [Wooldridge 2009]. In this context, the adoption of an Agent-Centered MAS (AC-MAS) revealed to be problematic, leading to the proposal of exploiting organizational aspects of such systems to promote coordination and cooperation towards the system's goals [Ferber et al 2004][Hübner et al 2002]. Following this trend, many organizational models were proposed to develop the so-called Organizational-Centered MAS (OC-MAS) [Dignum 2004][Ferber et al 2004][Hübner et al 2002], which were implemented based on a particular Organization Infrastructure (OI). However, this approach turned out to be restrictive, meaning that agents got heavily dependent on a particular OI. This raises the problem of agent interoperability among different organizations, which can be defined as the inability of agents designed to run in a particular organization to perform tasks in a different one. In this paper, we investigate the possibility of addressing this issue by extending the JaCaMo infrastructure [Boissier et al 2013], which already implements the Moise+ organizational model with artifacts, following the ORA4MAS approach [Hübner et al 2010] as an alternative to the OI, in order to incorporate other organizational models such as AGR [Ferber et al 2004] and OperA [Dignum 2004].

This article is organized as follows. Section 2 briefly explains the models and implementations upon which this paper is based. Section 3 presents a simplified model of Moise+ implementation using artifacts, and its proposed extension to include support to AGR and OperA, as a sequence of our previous work in [Muramatsu et al 2014]. Finally, in Section 4 we discuss some preliminary results about the extension.

2. Background: JaCaMo, ORA4MAS and NPL

2.1. JaCaMo

JaCaMo is a platform for programming MAS that combines agent, organization and environment programming into a single framework. This approach is possible due to integration of several technologies: (i) Jason [Bordini et al 2007], a platform for programming MAS using an agent-oriented approach; (ii) CArtAgO [Ricci et al 2007] for an environment-oriented approach; and (iii) Moise+ [Hübner et al 2002], for an organizational-oriented approach. In addition, ORA4MAS provides the means to integrate Moise+ organizational model through organizational artifacts provided by CArtAgO framework.

During runtime, agents in JaCaMo have direct access to the environment, by means of manipulating artifacts in the same way as described in CArtAgO. Concurrently, their actions are monitored by an organization, modeled in accordance to Moise+ and implemented in the environment as proposed by ORA4MAS. An important feature in JaCaMo, which will be explored in this work, is the presence of a Normative Programming Language (NPL) interpreter within the ORA4MAS organizational artifacts to run the organization. This implies that the organization description must be translated into NPL before it is loaded by the organizational artifacts.

2.2. ORA4MAS

Traditionally, the implementation of OI is based on an architecture composed of services and special agents, located in a layer inaccessible to ordinary agents and dependent on its underlying organizational model [Coutinho et al 2007][Hübner et al 2010]. Although this approach successfully achieves its goals of running an OI, it has the disadvantage to be too strict and inflexible, as agents are required to know how the OI in which they are running is structured.

ORA4MAS (Organizational Artifacts for Multiagent Systems) was proposed as a solution to the limits imposed by the previously mentioned approach. Its goal is to make organizations more flexible by implementing it in a layer accessible to agents, exploiting artifacts to deal with aspects of the organizational model. It is important to stress that ORA4MAS solution is to move the required knowledge of the OI from the agents to the organizational artifacts.

The role of artifacts in this context is to provide operations and information regarding the organization to any agent that participates in it. For instance, if an agent wants to adopt a role, it must trigger the correspondent operation on the artifact. Moreover, it can easily get information about the organization state (for example, the available roles) by inspecting the artifact's observable properties. Organizational agents are proposed in ORA4MAS to deal with aspects that require reasoning. For instance, whenever an agent triggers a forbidden operation, one of two actions can be taken: regimentation or sanction. Regimentation immediately blocks the operation and recovers the state of the system. Sanction involves notifying this organizational agent about the operation, who in turn decides on what action to take.

2.3. Normative Programing Language (NPL)

ORA4MAS organizational artifacts have embedded within them an NPL engine in order to execute its functions. This language is, as the name implies, a programming language based on norms. In general, norms are composed of a statement describing an expected pattern of behavior and the consequences of disregarding it. This way, each norm can be considered an obligation to all agents it is imposed upon. In addition, the language utilizes facts, which are statements of information, and inference rules in order to track the system state. Therefore, NPL programs are composed of facts and inference rules plus a set of norms.

Utilizing the OM translations into NPL from our previous work, it should be possible to have JaCaMo working with organizational models other than Moise+. The next step is to conceptualize the structure of the organizational artifacts needed to run these normative programs, to reflect the features of each organizational model. Section 3 describe the process of modifying JaCaMo framework.

3. Extending JaCaMo

In this section, we will explore how it is possible to create a set of organizational artifacts to support AGR and OperA, based on the existent implementation of Moise+ artifacts. The idea is to check the possibility to extend JaCaMo with these artifacts, making it capable to run multiple organizations and have agents to interoperate among them. Our analysis of the Moise+ artifacts resulted in the simplified class diagram shown in Figure 1.

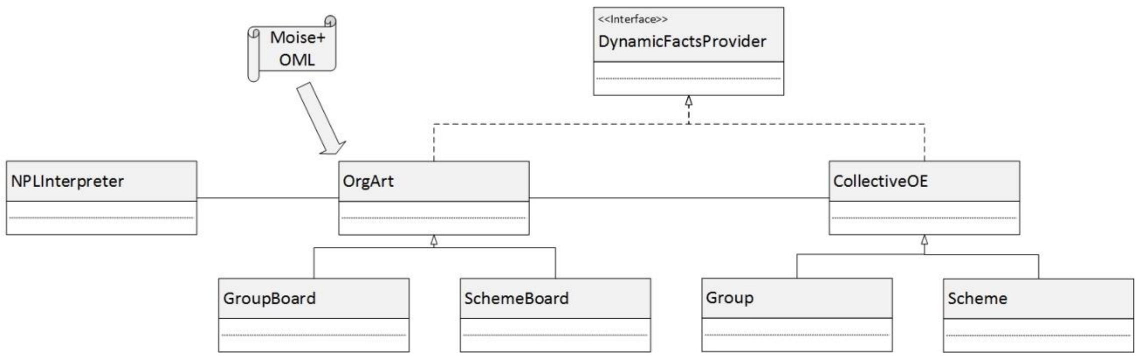

Figure 1: Class diagram showing a simplified structure of Moise+ artifacts

We notice from the class diagram that the organizational artifacts for Moise+ consist of two artifacts, one for each organizational dimension, modelled by the `GroupBoard` and `SchemeBoard` classes. Both artifacts inherit from the `OrgArt` class, which defines the general structure of an organizational artifact. As mentioned before, a normative engine is embedded in these artifacts to effectively run the organization in terms of norms. That's the role of the `NPLInterpreter` class, which compounds the structure of `OrgArt` and is triggered by any action to verify compliance with norms. This is the reason why there is no artifact dedicated to the Moise+'s normative dimension. Furthermore, the `CollectiveOE` class is also an important component of `OrgArt`. At runtime, several actions take place in an organization, leading to the generation of many dynamic facts. This class is responsible to define and store these dynamic facts, and answer to any queries regarding them. The `Group` and `Scheme` classes are specializations of this class, defining constructs specific to the `GroupBoard` and

SchemeBoard artifacts, respectively. Finally, the DymanicFactsProvider interface provides a model for the OrgArt and CollectiveOE classes to respond to the aforementioned queries related to dynamic facts.

In the current implementation, the OrgArt artifact receives a Moise+ description in its Organizational Modeling Language (OML). However, as we want organizational artifacts to deal with models other than Moise+, we have modified OrgArt to accept a description in NPL. Moise+ will remain using the original OrgArt, while other models will use this new version.

3.1. AGR

AGR is a simple organizational model that focuses on the concepts of Agent, Group and Role, as its acronym suggests. It allows the designer to define some constraints upon the structure of the organization (in terms of groups and roles, similarly to Moise+) and upon the communication of agents (in terms of interactions). A detailed description of this model can be found in [Ferber et al 2004].

A natural approach to convert the structure shown in Figure 1 to AGR might consist in adapting the GroupBoard, due to its similarity to the structural dimension of AGR, removing the SchemeBoard, since there is no such notion in this model, and creating a CommunicationBoard to deal with the communication constraints.

However, since this model restricts interactions to be placed only between agents enacting roles in the same group, it seems more convenient to implement this portion of the organization also in the GroupBoard. Therefore, we believe that a possible structure of AGR would consist only in a GroupBoard artifact, with the correspondent Group class with all the dynamic facts specified. Of course, proper norms dealing with AGR constructs should be loaded in this artifact's normative engine for it to work as predicts its model. Figure 2 illustrates the proposed changes for AGR.

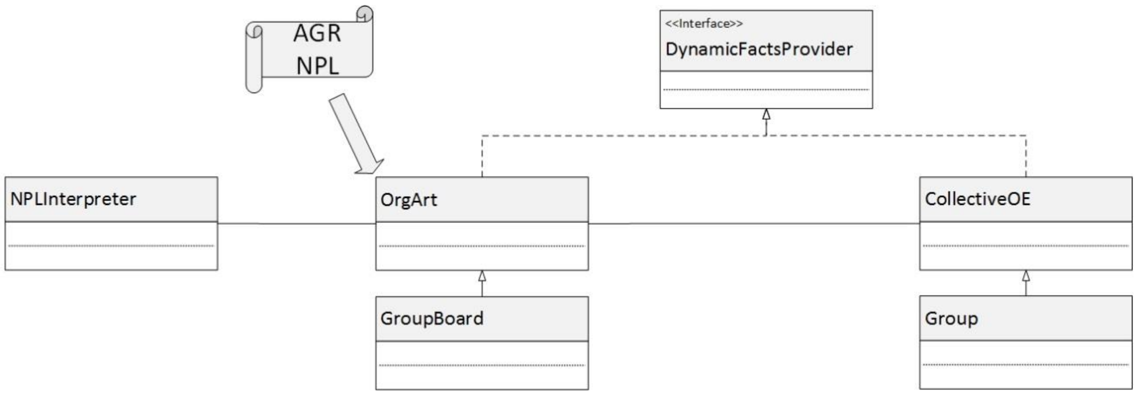

Figure 2: Class diagram showing a simplified structure of AGR artifacts

3.2. OperA

The OperA organizational model was designed to provide interaction and collaboration among agents immersed in the system without compromising autonomy between society design and agent design.

Due to the differences between Moise+ and OperA organizational model, several modifications have to be made in order to take the existing Moise+ artifacts' structure and transform it into one suitable for the OperA model.

One of the biggest changes is to the `GroupBoard` artifact as OperA's structural specification is not group centered as Moise+ or AGR are. However, most of the current functionality of the artifact should be preserved as there are several other structural characteristics shared between the models. Another major change is the replacement of the `SchemeBoard` by `SceneBoard` in order to implement OperA's functional structure based on scenes rather than schemes as Moise+ is.

These modifications to `GroupBoard` and `SchemeBoard` also imply that the `Group` and `Scheme` classes are not relevant to the model and new classes have to be created to provide the necessary constructs for the new artifacts. Finally, a `CommunicationBoard` class has to be implemented to deal with the communication aspects. Figure 3 illustrates the proposed changes for OperA.

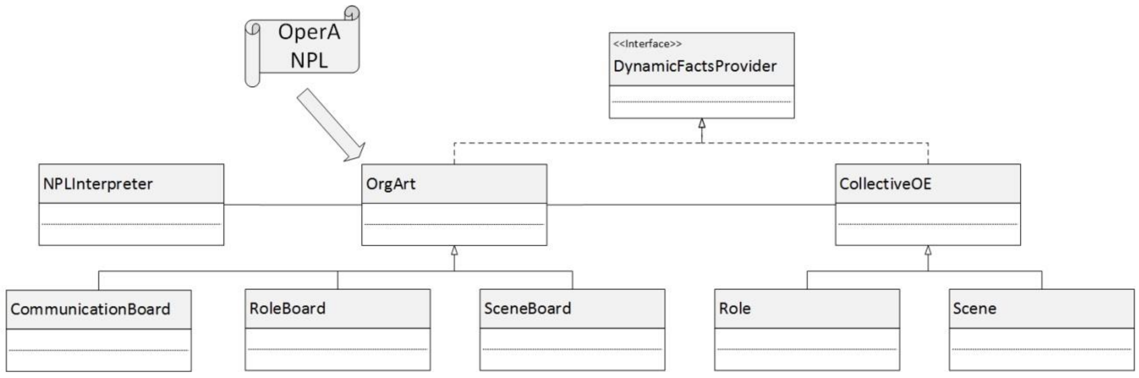

Figure 3: Class diagram showing a simplified structure of OperA artifacts

4. Discussion

Currently, our work is focused in analyzing at what extent we must change the artifacts' code in order to make it accept AGR or OperA. Fortunately, it is getting clear that thanks to the normative engine, organizational artifacts are not deeply dependent on Moise+ constructs, since they work upon the primitives of norms, facts and rules that are common to any normative program describing an organization model.

In the case of AGR, some changes in the `GroupBoard` have already been made to remove unused concepts (related schemes and sub-groups, for instance) and to add new ones, such as the notion of group structures and the dependence and correspondence rules between roles. As for OperA, the `GroupBoard` class was removed and some of its functions have been moved to another artifact (`RoleBoard`) that does not include the group concept. However, our work is still ongoing in these implementations.

Once we have completely developed these artifacts, we believe it should be possible to build MAS in which a single agent can interact with multiple organizations. This is because agents are no longer required to comprehend the organization structure, thus making their design independent of organizational models. The only knowledge agents are required to have in order to interact with any organization is how to handle CArtAgO artifacts and how to deal with obligations.

5. Conclusion

In this paper we presented an approach to provide organization interoperability by extending JaCaMo with the inclusion of other organizational artifacts. It is still ongoing work, since coding is at its very beginning. Nevertheless, since it is strongly based on the idea already implemented within JaCaMo for the Moise+ organizational model, we believe that we will succeed.

6. Acknowledgements

Tomas M. Vitorello is partially supported by grant 013/17973-1, São Paulo Research Foundation (FAPESP). Fabio T. Muramatsu is partially supported by grant #013/17948-7, São Paulo Research Foundation (FAPESP). Anarosa A. F. Brandão is partially supported by grant #010/2640-5, São Paulo Research Foundation (FAPESP).

7. References

[Boissier et al 2013] Boissier, O.; Bordini, R.; Hübner, J.; Ricci, A.; Santi, A. Multi-agent oriented programming with JaCaMo, Science of Computer Programming 78 (2013) 747–761.

[Bordini et al 2007] Bordini, R.; Hübner, J.; Wooldridge, M.; Programming Multi-Agent Systems in AgentSpeak using Jason, Wiley, 2007.

[Coutinho et al 2007] Coutinho, L.R.; Brandão, Anarosa A.F. ; Sichman, J.S. ; Boissier, O. . Organizational Interop-erability in Open Multiagents Systems - An Approach based on Metamodels and Ontologies. In: Proc.of the 2nd Workshop on Ontologies and Metamodels in Software Engineering WOMSDE 2007, 2007.

[Dignum 2004] Dignum. V. A Model for Organizational Interaction: based on Agents, founded in Logic. SIKS Dissertation Series 2004-1. SIKS, 2004. PhD Thesis.

[Ferber et al 2004] Ferber, J., Gutknecht, O., Michel, F.: From agents to organizations: an organizational view of multi-agent systems. In: AOSE'03. Volume 2935 of LNCS., Springer (2004) 214–230.

[Hübner et al 2002] Hübner, J.F., Sichman, J.S., Boissier, O.: A model for the structural, functional, and deontic specification of organizations in multiagent systems. In: SBIA'02: 16th Brazilian Symposium on Artificial Intelligence. Volume 2507 of LNAI., Springer (2002) 118– 128.

[Hübner et al 2010] Hübner, J., Boissier, O., Kitio, R., Ricci, A., Instrumenting multi-agent organisations with or-ganisational artifacts and agents. In Auton Agent Multi-Agent Syst (2010) 20:369–400.

[Muramatsu et al 2014] Muramatsu, F.T, Vitorello, T.M and Brandão, A.A.F. Towards organizational interoperability through artifacts. In Proceedings of Environments for Multiagent Systems 10 years later - E4MAS 2014 (to appear)

[Ricci et al 2007] Ricci, A., Viroli, M., Omicini, A., CArtAgO: A Framework for Prototyping Artifact-Based En-vironments in MAS. In D. Weyns, H.V.D. Parunak, and F. Michel (Eds.): E4MAS 2006, LNAI 4389, pp. 67–86, 2007. Springer-Verlag Berlin Heidelberg 2007.

[Wooldridge 2009] Wooldridge, M. An Introduction to MultiAgent Systems, John Wiley & Sons, 2009.