An Organisation Infrastructure for Multi-Agent Systems based on Agent Coordination Contexts

Mirko Viroli, Alessandro Ricci, Andrea Omicini
DEIS, Alma Mater Studiorum – Università di Bologna a Cesena
Via Venezia 52, 47023 Cesena, Italy
{mviroli,aricci,aomicini}@deis.unibo.it

ABSTRACT

We present an organisation infrastructure for open Multi-Agent Systems built upon a role-based access control model (RBAC), which provides agents with means to enable and control actions toward the other entities in the MAS (agents and resources).

In particular, an Agent Coordination Context (ACC) is created and assigned to an agent as it enters the MAS, which acts as a sort of private interface for the agent towards the environment: any agent action is thereafter enabled and controlled by that ACC, which prevents those behaviours that are incorrect with respect to the role(s) played by the agent. To this end, each role is assigned a policy that flexibly specifies possible/allowed actions and perceptions over time.

Categories and Subject Descriptors

D.2.10 [**Software Engineering**]: Design; I.2.11 [**Distributed Artificial Intelligence**]: Multi-agent Systems

General Terms

Languages, Design

Keywords

Agent Coordination, Agent Infrastructures, Role-Based Access Control

1. FROM RBAC TO RBAC-MAS

In RBAC, a *role* is a semantic construct around which access control policy is formulated, bringing together a given collection of users and permissions in a transitory way [5]. The role concept assumes several manifestations in the literature, which RBAC aims to accommodate and capture. On the one hand, a role can represent *competency* to do specific tasks, but also the embodiment of *authority and responsibility*: all these cases are expressed in terms of enabled/disabled operations over objects in the organisation. These inter-role relations can be used to enforce security policies that include e.g. *separation of duties*, *least privilege*, and *data abstraction*. The reference architecture formally defined in [1] is pictorially reported in Figure 1.

The MAS perspective introduces a new view over interaction, and therefore over organisation; as a result, RBAC has to be suitably adapted and extended to suit the nature of the agent-oriented abstractions. Accordingly, we introduce a new model for RBAC-like organisation of MAS called RBAC-MAS, depicted in Figure 2, featuring the following main peculiarities with respect to standard RBAC:

Agent Classes. Instead of RBAC *users*, RBAC-MAS has *agent classes*. As an agent enters in the MAS it should be recognised as belonging to precisely one of these classes — most likely, after an authentication process.

Actions and Perceptions. Operations and objects are strictly related in MAS, and moreover, agent interactions are typically structured in terms of actions and perceptions.

Therefore RBAC-MAS introduces the notions of *action* and *perception*, which are seen as *operations* involving a given *object*.

Policies. Instead of *permissions* (subsets of operations over objects), RBAC-MAS introduces *policies*, which are protocols (possibly infinite, concurrent, and non-deterministic) of actions and perceptions.

2. INFRASTRUCTURE SUPPORT AND ACCS

Based on this general RBAC-MAS model, we develop an infrastructure approach to organisation in MAS. In particular, we provide concepts and design of an infrastructure that can be exploited to support at run-time all the

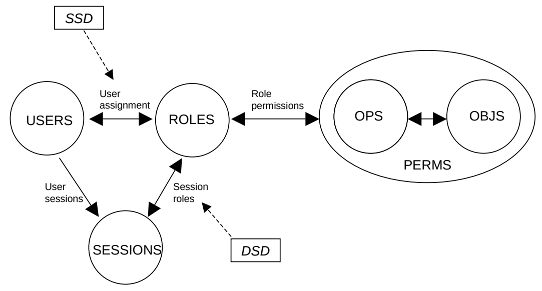

Figure 1: RBAC Reference Model

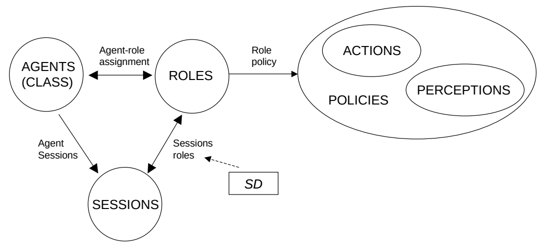

Figure 2: RBAC-MAS Reference Model

constructs and components of RBAC-MAS, including roles, agent classes, policies, and sessions, as well as all their relationships.

2.1 Sessions and ACCs

A key notion introduced in RBAC models is that of session. As an agent enters a MAS it is given a session, which the agent uses to activate and deactivate roles, and which is used to check whether agent actions are correct with respect to the roles played. Thus, because sessions have a strong effect on the run-time behaviour of agents in MASs, we focus our infrastructure approach on the idea of exploiting a run-time abstraction in charge of realising an agent session. To this end, we use the Agent Coordination Context (ACC) notion, introduced in a general form in[2] and exploited in a specific fashion in [4] as a means to enable/control agent interactions in TuCSoN (`http://tucson.sourceforge.net`).

Generally speaking, an ACC represents the conceptual boundary between the agent and the environment, encapsulating the environment *interface* to be used by the agent. Being an interface, the ACC both (i) works as a model for the agent environment, and (ii) enables and rules the interactions between the agent and the environment. Hence, the ACC abstraction is particularly fruitful to model the presence of an agent within an organisation, by defining its admissible interactions with respect to organisation resources and its admissible communications toward the other agents belonging to the organisation. Thus, an ACC is meant to enact the policies assigned to an agent — that is, to physically allow admissible agent actions/perceptions toward the environment — and to make them available for agent run-time inspection and, possibly, meta-level reasoning.

2.2 Agent Interaction with ACCs

Two basic stages characterise the life-time of an ACC: *ACC negotiation* and *ACC use*. In order to take part actively to an organisation, to access its objects (other agents or resources), an agent must first *negotiate* an ACC with the organisation infrastructure.

In particular, through a standard welcome service of the infrastructure, an agent can authenticate its identity and therefore receive its own private ACC, representing its working session in the current MAS. This step can actually feature a number of details that we here abstract away from, concerning both the authentication process and how/where locating the ACC prepared for the agent. Receiving an ACC does not conclude negotiation: the ACC is initially void, for no actions have been enabled, yet. The agent should request to the ACC the activation of some role, which is granted only if the agent/role permissions allow this. If activation succeeds, the agent can start playing the role by executing actions according to the policies specified for that role. If all the policies associated to a given role are terminated, the role can be deactivated. Hence, note that the dynamics of role activation and deactivation is not fixed a priori, but varies depending on the agent needs. When no roles are activated the agent can leave the ACC, therefore leaving the whole MAS — possibly entering later again with a new authentication. Failing to deactivate a terminated role or leaving a void ACC is prevented where possible, or can be tracked and considered as an agent violation in other cases — e.g. involving legal consequences.

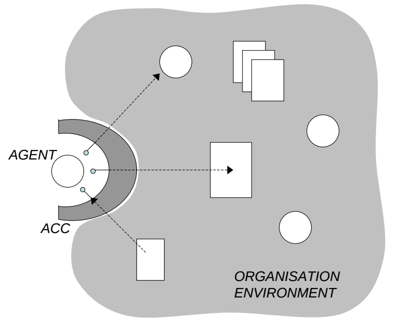

Figure 3: ACC as a Control Room

Besides these negotiation aspects, the agent typically use an ACC by executing actions (receiving perceptions) according to the currently activated roles. In particular, the role is initially associated a number of policies, each representing a protocol of actions and perceptions. Whichever language for policies is used — see e.g. [3] — it should describe what are the actions/perceptions allowed at a given time, and what is the next state of policies as one action is executed or a perception occurs. So, an agent can executed any action (received a perception) allowed by any current policy of any currently activated role.

Our current research efforts focuses on implemented this organisational support on top of a new release of TuCSoN.

3. REFERENCES

[1] D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli. Proposed NIST standard for role-based access control. *ACM Transactions on Information and System Security (TISSEC)*, 4(3):224–274, 2001.

[2] A. Omicini. Towards a notion of agent coordination context. In *Process Coordination and Ubiquitous Computing*, pages 187–200. CRC Press, 2002.

[3] A. Omicini, A. Ricci, and M. Viroli. Formal specification and enactment of security policies through Agent Coordination Contexts. In *Proceedings of SecCo*, volume 85(3) of *ENTCS*. Elsevier Science B. V., 2003.

[4] A. Ricci, M. Viroli, and A. Omicini. Agent coordination context: From theory to practice. In *Cybernetics and Systems 2004*, volume 2, pages 618–623. Austrian Society for Cybernetic Studies, 2004.

[5] R. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based control models. *IEEE Computer*, 29(2):38–47, 1996.