Normative Ontologies to Define Regulations Over Roles in Open Multi-Agent Systems

Carolina Felicíssimo, Carlos Lucena, Gustavo Carvalho and Rodrigo Paes

Departamento de Informática – PUC–Rio
Rua Marquês de São Vicente 225, Gávea – 22.451-900 – Rio de Janeiro – RJ – Brazil
{cfelicissimo, lucena, guga, rbp}@inf.puc-rio.br

Introduction

Roles are being represented as a fundamental concept in most knowledge representation languages, mainly in the design of open software systems. Unknown entities playing their roles in such systems increase the opportunity to happen unpredictable and non-desired situations. Because of that, regulations over roles are necessary.

In open multi-agent systems, it is required to have mechanisms to guide the behaviors of agents, especially when their behaviors affect other agents and the agents playing in it are unknown. Norms, characterized by their *prescriptiveness*, *sociality* and *social pressure* [López et al., 2005], can police agents' actions as a regulatory mechanism.

This paper presents norms and ontologies' definition for actions performed by agents. This approach defines that regulations have a semantic support provided by an ontology, specially designed and built for it. This ontology is a generic one and need to be extended for specific domains. An ontology for the urban traffic domain was extended and instantiated in an urban traffic simulator system, developed as a case study. The generic and extended ontologies are also presented in this paper.

Open Multi-Agent Systems

Agents are, in short, goal-oriented entities [Jennings, 2000]. Multi-agent systems (MAS) [Weiss, 1999] are often understood as complex systems where a multitude of agents interact, usually with some intended individual or collective purpose, and react to changing environments as adaptive entities. In open MAS, agents are also self-governed autonomous entities that pursue their own individual goals based only on their own beliefs and capabilities [Abdelkader, 2003].

Openness has led to software systems that have no centralized control and that are composed of autonomous entities [Agha, 1997], as agents. In open MAS, agents need to be autonomous normative entities, able to take into account the existence of social norms in their decisions (either to follow or violate a norm) and able to react to norms violations by other agents [Castelfranchi et al., 1999].

In the scenario where unknown agents are playing without a centralized control, the chaos open MAS can become has to be avoid. The use of regulatory mechanisms to control the behavior of agents is suggested, but the construction of open normative MAS is still a great challenge.

Supports for a Regulatory Mechanism

In an open environment, what agents will play in it is unpredictable and the internal aspects of the agents from there are inaccessible. Because of those characteristics, regulatory mechanisms for open MAS have to enforce agent roles instead of agent instances and must be performed by the environment where agents will play. Regulatory mechanisms cannot simply be implicitly represented by constraints in agents' architectures or external fixed rules.

Norms, a regulatory mechanism, can control the actions performed in an open MAS defining which agents are permitted, obligated and prohibited to execute those actions. A <u>permitted</u> norm defines that an act <u>is allowed to be performed</u>; an <u>obligatory</u> norm defines that an act <u>must be performed</u>; a <u>prohibited</u> norm defines that an act <u>must not be performed</u>. The three types of norms described represent the three fundamental *deontic statuses* of an action [Alberti et al., 2005] from Deontic Logic [Wright, 1951]. Deontic Logic enables to address the issue of explicitly and formally defining norms and dealing with their possible violation.

The three basic deontic notions of permission, obligation, and prohibition are logically connected as presented by the following statements [Wright, 1951]:

- If an act <u>is permitted</u>, then it <u>is not prohibited</u>.
- If an act <u>is obligatory</u>, then it <u>is permitted</u> and it <u>is not prohibited</u>.
- If an act <u>is prohibited</u>, then it <u>is not obligatory</u> and it <u>is not permitted</u>.

- If is not permitted not perform an act, then this act is obligatory.
- If is prohibited not perform an act, then this act is obligatory.
- If is permitted or is obligatory not perform an act, then this act is prohibited.
- If is permitted perform and not perform an act, then this act is permitted.

Normative Ontologies

To provide consciousness of agents on MAS, a semantic support is desired. This type of support can be given by ontologies, making the represented information of a domain easier for machines to automatically process their meaning [McGuinness and Harmelen, 2004]. Ontologies are conceptual models that embody shared conceptualizations of a given domain [Gruber, 1993]. Ontologies languages are designed to be used by applications (machines) that need to process the content of information instead of just presenting information to humans [Smith et al., 2004].

An ontology can support norms regulation according to the Deontic Logic. An example of a generic normative ontology based on roles was built in the Protégé ontology editor [Protégé, 2005] and is illustrated in Fig.1, by using the Ontoviz plug-in [Ontoviz, 2005]. The generic ontology needs to be extended for specific domains. The normative generic ontology has five related main concepts, all in the same hierarchical level (Fig. 1): Role, Norm, Penalty, Action and Place.

The **Role** concept has the two attributes: *hasNorm*, that holds one-to-various Norms instances to represent the role's norms, and *playIn*, that holds one Place instance to represent where the role's norms are valid.

The **Norm** concept has the three attributes: *isActive*, that holds one Boolean value as a flag to represent if the norm is active or not, *hasPenalty*, that holds one Penalty instance to represent the norm's penalty, and *regulate*, that holds the regulated Action instance; and the three sub-concepts: **Permission**, **Obligation** and **Prohibition**.

The **Penalty** concept has an integer value attribute called *hasFine* to represent its fine and the three sub-concepts: **PermissionPenalty**, **ObligationPenalty** and **ProhibitionPenalty**. The Penalty concept is used to intimidate norms violation.

The **Action** and **Place** concepts don't have attributes specified in the generic ontology.

Normative ontologies provide information for norm-autonomous agents committed with their roles base their behavior on, for example, goals and plans. However, viruses can inhabit normative environments and are examples of agents non-compliant with norms, i.e. norm violation entities, which are compliant just with their own goals and plans.

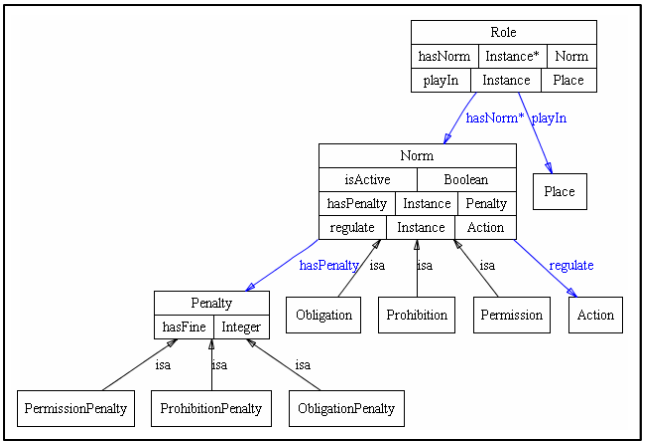

Fig.1. A generic normative ontology based on roles

In the presented normative ontology, each agent's role has associated with it, norms and the place where those are valid. An agent role's norm regulates an action and has an associated penalty to inhibit norm violation. With this design, the generic normative ontology permits to regulate agents' actions by just letting agents to be committed with the normative roles from the MAS.

Norm-autonomous agents capable to obtain the semantic support provided (their internal structures access ontologies) are agents with intelligent for norm violation by an action selection mechanism. Action selection mechanisms [Tyrrell, 1993; Blumberg, 1997] permit effectiveness for agents achieve their goals.

Case Study

Urban Traffic Simulator System – UTSS – is an open multi-agent system with a semantic support to the governance of norms. UTSS represents an urban environment that simulates the behavior of three agent's roles: car driver, police officer and pedestrian. These three roles are played by active entities. Besides, there are other elements as traffic signs, urban paths and places.

An urban traffic scenario is illustrated in Fig.2. The proposed environment is composed (i) by agents playing the car driver role, the pedestrian role and/or the police officer role; (ii) by four types of traffic signs (traffic signals, speed limit signs, one-way signs and two-way signs); (iii) by the five illustrated numbered urban paths and (iv) by the four places: church (accessed via path 1), house in the town (accessed via paths 2 and 4), house on the mountains (accessed via path 3) and hospital (accessed via path 5).

The paths from the proposed environment have different characteristics, depending on where they are located. Paths 1 and 2 are located in a street called *Toneleros Street*, paths 3 and 4 are located in an avenue called *Sernambetiba Avenue* and path 5 is an emergency exit. Streets and avenues can have different meanings for their traffic signs.

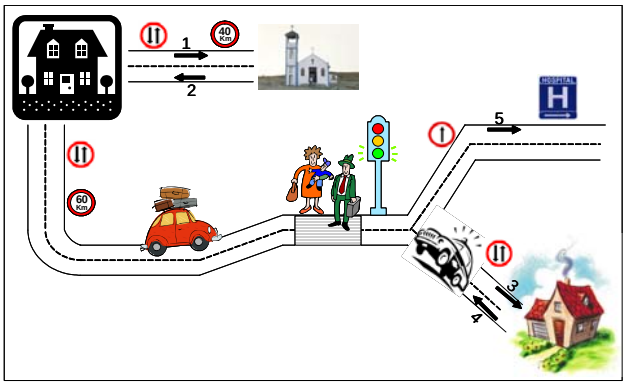

Fig.2. An urban traffic's scenario

Car drivers and pedestrians go and come from different places by using urban paths, which have several traffic signs. According to their goals, car drivers and pedestrians choose whether to obey or not the traffic sign rules. Police officers monitor the laws of traffic signs and can apply penalties for disobedient active entities. A traffic sign can have zero or many police officers monitoring it.

Ontology Extension

For the case study, the generic normative ontology presented, illustrated in Fig.1, was extended by its concepts **Role**, **Action** and **Place**. The extended ontology is called UTSS's ontology and is illustrated in Fig.3.

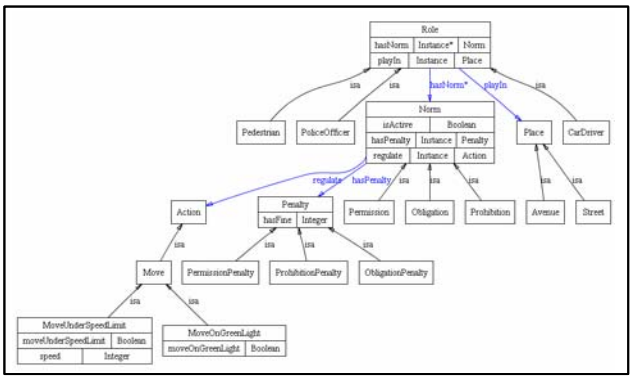

Fig.3. A normative ontology for the urban traffic domain

The Role concept was extended by adding the three sub-concepts: **CarDriver**, **Pedestrian** and **PoliceOfficer**. Those concepts represent specific roles from the urban traffic domain.

The Action concept was extended to have the **Move** sub-concept with its two sub-concepts: MoveUnderSpeedLimit and MoveOnGreenLight. The **MoveUnderSpeedLimit** concept has one Boolean value attribute called *moveUnderSpeedLimit* and one integer value attribute called *speed*. The **MoveOnGreenLight** concept has one Boolean value called *moveOnGreenLight*.

The Place concept was extended by adding the two sub-concepts: **Avenue** and **Street**. Those types of places are specific urban paths' concepts.

Ontology Instantiation

The UTSS's ontology was instantiated by its classes: *Avenue*, *Street*, *CarDriver*, *Permission*, *Obligation*, *Prohibition*, *PermissionPenalty*, *ObligationPenalty*, *ProhibitionPenalty*, *MoveOnGreenLigth* and *MoveUnderSpeedLimit*.

According to the ontology model illustrated in Fig.3, each role instance has norms valid in a place. For the case study presented, the Place concept was instantiated by its sub-concepts Avenue and Street. The *Sernambetiba Avenue* and the *Toneleros Street*, illustrated in Fig.4, represents the place instances created.

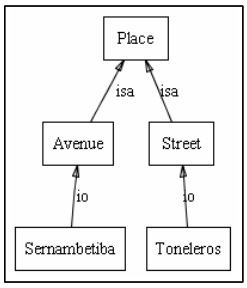

Fig.4. Example of Avenue and Street instances

For the car driver concept, two instances were created with the identifiers: *MathewCarDriver* and *BobCarDriver*. Fig.5 illustrates the roles instances created.

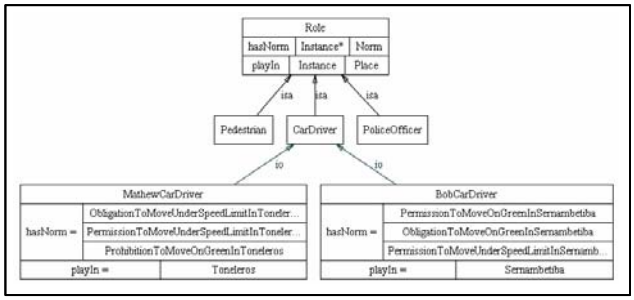

Fig.5. Example of Role instances

In the UTSS's ontology, roles have norms to regulate actions. Fig.6 illustrates the *BobCarDriver* role instance, playing in the *Sernambetiba* place instance, with its norms to regulate the *MoveOnGreenInSernambetiba* action instance. *BobCarDriver* is regulated by the *PermissionToMoveOnGreenInSernambetiba* and the *ObligationToMoveOnGreenInSernambetiba* norms instances with their respectively penalties instances, *PermissionPenaltyToMoveOnGreenInSernambetiba* and *ObligationPenaltyToMoveOnGreenInSernambetiba*.

The *MoveOnGreenInSernambetiba* action, illustrated in Fig.6, has its *moveOnGreenLight* attribute set with the **true** value, meaning that the *BobCarDriver* has the permission (*PermissionToMoveOnGreenInSernambetiba*) and the obligation (*ObligationToMoveOnGreenInSernambetiba*) to move on the green light in the *Sernambetiba Avenue*. If agents playing the *BobCarDriver* role don't obey the law, other agents playing the police officer role can apply the

PermissionPenaltyToMoveOnGreenInSernambetiba and the *ObligationPenaltyToMoveOnGreenInSernambetiba* penalties, totalizing a fine with its value equal to **15** (**5** from the first penalty and **10** from the second one). Agents playing the car driver role and with a high fine value can be banish from the environment.

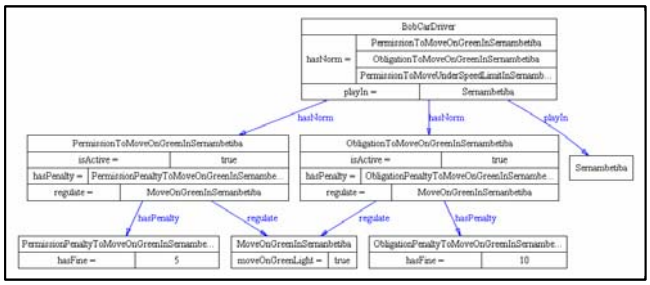

Fig.6. *BobCarDriver* with its norms regulating the *MoveOnGreenInSernambetiba* action

The *MoveUnderSpeedLimitInSernambetiba* action, illustrated in Fig.7, is regulated by the *PermissionToMoveUnderSpeedLimitInSernambetiba* norm, which has the *PermissionPenaltyToMoveUnderSpeedLimitInSernambetiba* penalty with its fine attribute's value equal to **1**. The action has its *moveUnderSpeedLimit* attribute set to **false** and its *speed* attribute set to **60**, meaning that agents playing the role have the permission to move with their speed **greater** than **60** (the speed limit specified). This is an example where the speed limit sign addresses the minimum speed instead of the maximum speed.

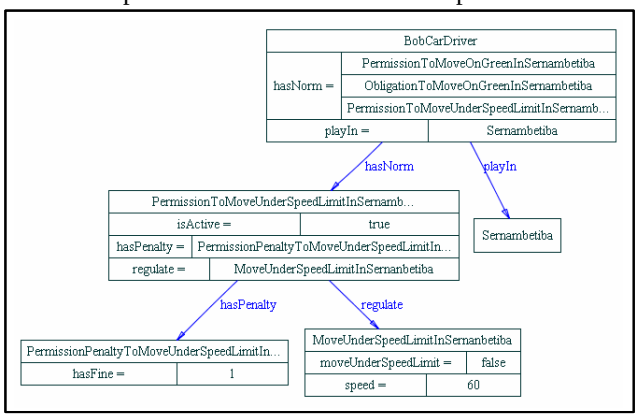

Fig.7. *BobCarDriver* with its norms regulating the *MoveUnderSpeedLimitInSernambetiba* action

Another example of an action regulated by norms is illustrated in Fig.8. The *MathewCarDriver* role instance has the *ObligationToMoveUnderSpeedLimitInToneleros* and the *PermissionToMoveUnderSpeedLimitInToneleros* norms to regulate the *MoveUnderSpeedLimitInToneleros* action. The action has its *moveUnderSpeedLimit* attribute set to **true** and its *speed* attribute set to **40**, meaning that disobedient agents playing the role, i.e. moving with their speeds **greater** than 40 (speed limit specified), will totalize a fine with its value equal to **10** (**7** from the *ObligationPenaltyToMoveUnderSpeedLimitInToneleros* penalty and **3** from the *PermissionPenaltyToMoveUnderSpeedLimitInToneleros* penalty). The *ProhibitionToMoveOnGreenInToneleros* norm instance prohibits agents playing the *MathewCarDriver* role to execute the *MoveOnGreenInToneleros* action in *Toneleros Street* because there doesn't exist any traffic signal (check paths 1 and 2, from the *Toneleros Street*, illustrated in Fig.2).

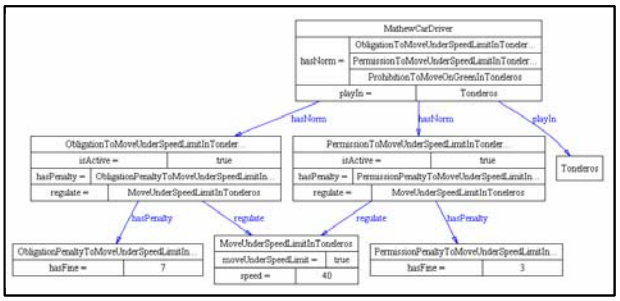

Fig.8. *MathewCarDriver* with its norms regulating the *MoveUnderSpeedLimitInToneleros* action

Conclusion

In open MAS, agents play without a centralized control. To minimize the chaos that open MAS can become, mechanisms to regulate the actions permitted, obligated and prohibited to be performed by agents playing their roles are desired.

This paper presents a normative approach that uses an ontology to define regulations over roles in open MAS. The ontology is independent of domain and it has five related basic concepts: Role, Norm, Penalty, Action and Place. This ontology's structure provides a semantic support for agents to base their behavior according to norms and to reason about action selection.

The generic normative ontology based on roles presented was instantiated and extended for a case study of the urban traffic domain. As a future work, case studies from other domains are going to be developed. The results of others case studies will probably interfere in the current version of the generic ontology, which will probably need to be reviewed and also improved.

References

Abdelkader, G. *Requirements for achieving software agents autonomy and defining their responsibility*. In Proc. Autonomy Workshop at AAMAS, 2003.

Agha, G. A. *Abstracting Interaction Patterns: A Programming Paradigm for Open Distributed Systems*. In (Eds) E. Najm and J.-B. Stefani, Formal Methods for Open Object-based Distributed Systems IFIP Transactions, Chapman & Hall, 1997.

Alberti, M.; Gavanelli, M.; Lamma, E.; Mello, P.; Torroni, P.; Sartor, G. *Mapping Deontic Operators to Abductive*

Expectations. In Proc. of the 1st International Symposium on Normative Multiagent Systems (NorMAS 2005), AISB 2005, Hertfordshire, Hatfield, UK, April 2005.

Blumberg, B. M. *Old Tricks, New Dogs: Ethology and Interactive Creatures*. PhD Thesis from the program in media arts and sciences, school of architecture and planning, Massachusetts Institute of Technology. February, 1997.

Castelfranchi, C.; Dignum, F.; Jonker, C.M.; Treur, J. *Deliberative Normative Agents: Principles and Architecture*. In Proc. of the Sixth International Workshop on Agent Theories, Architectures, and Languages (ATAL-99), Orlando, FL, July 15-17, 1999.

Gruber, T. R. *A translation approach to portable ontology specifications*. Knowledge Acquisition, Vol. 5, Issue 2 (1993, June), pages: 199-220, ISSN:1042-8143.

Jennings, N. R. *On agent-based software engineering*. In Artificial Intelligence 117 (2000), pp. 277-296, Elsevier Press, April, 2000.

Lopéz, F. L. y; Luck, M.; d'Inverno, M. *A Normative Framework for Agent-Based Systems*. In Proc. of the 1st International Symposium on Normative Multiagent Systems (NorMAS 2005), AISB 2005, Hertfordshire, Hatfield, UK, April 2005.

McGuinness, D. L.; Harmelen, F. van. *OWL Web Ontology Language Overview*. W3C Recommendation 10 February 2004. In <http://www.w3.org/TR/owl-features/>. Accessed in May, 2005.

Ontoviz. Allows to visualize Protégé ontologies with the help of a highly sophisticated graph visualization software. In <http://protege.stanford.edu/plugins/ontoviz/ontoviz_tab_bin.zip/>. Accessed in May, 2005.

Protégé. A free, open source ontology editor and knowledge-base framework. Current recommended release: version 3.0. In <http://protege.stanford.edu/download/release/full/>. Accessed in May, 2005.

Smith, M. K.; Welty, C.; McGuinness, D. L. *OWL Web Ontology Language Guide*. W3C Recommendation 10 February 2004. In <http://www.w3.org/TR/owl-guide/> . Accessed in May, 2005.

Tyrrell, T. *Computational Mechanisms for Action Selection*. PhD Thesis from University of Edinburgh. 1993

Weiss, G. *Multi-agent systems*. MIT Press. 1999.

Wright, G. H. von. *Deontic Logic*. Mind, New Series, Vol. 60, No. 237 (Jan., 1951), pages. 1-15.

We gratefully acknowledge the financial support provided by the CNPq as part of individual grants and of the ESSMA project (552068/2002-0).