PUC

ISSN 0103-9741

Monografias em Ciência da Computação
n° 05/2018

Extending BDI Multiagent Systems with Agent Norms

Francisco José Plácido da Cunha
Tassio Ferenzini Martins Sirqueira
Marx Leles Viana
Carlos José Pereira de Lucena

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
RUA MARQUÊS DE SÃO VICENTE, 225 - CEP 22451-900
RIO DE JANEIRO - BRASIL

Monografias em Ciência da Computação, No. 05/2018 ISSN: 0103-9741
Editor: Prof. Carlos José Pereira de Lucena July, 2018

Extending BDI Multiagent Systems with Agent Norms

Francisco José Plácido da Cunha, Tassio Ferenzini Martins Sirqueira, Carlos José Pereira de Lucena

Departamento de Informática – Pontifícia Universidade Católica do Rio de Janeiro (PUC Rio)
{fcunha, tmartins, mleles, lucena}@inf.puc-rio.br

Abstract. Open Multiagent Systems (MASs) are societies in which heterogeneous and independently designed entities (agents) work towards similar, or different ends. Software agents are autonomous and the diversity of interests among different members living in the same society is a fact. In order to deal with this autonomy, these open systems use mechanisms of social control (norms) to ensure a desirable social order. This paper considers the following types of norms: (i) *obligation* — agents must accomplish a specific outcome; (ii) *permission* — agents may act in a particular way, and (iii) *prohibition* — agents must not act in a specific way. All of these characteristics mean to encourage the fulfillment of norms through rewards and to discourage norm violation by pointing out the punishments. Once the software agent decides that its priority is the satisfaction of its own desires and goals, each agent must evaluate the effects associated to the fulfillment of one or more norms before choosing which one should be fulfilled. The same applies when agents decide to violate a norm. This paper also introduces a framework for the development of MASs that provide support mechanisms to the agent's decision-making, using norm-based reasoning. The applicability and validation of this approach is demonstrated applying a traffic intersection scenario.

Keywords: BDI Agent, BDI4JADE Framework, Multiagent System, Normative Agent.

Resumo. Sistemas Multiagente (SMA) são sociedades nas quais entidades (agentes) heterogêneas e projetadas de maneira independente trabalham com fins similares ou diferentes. Os agentes de software são entidades autônomas e a diversidade de interesses entre os diferentes membros que vivem em uma mesma sociedade é um fato. Para lidar com tal autonomia, esses sistemas usam mecanismos de controle social (normas) para garantir uma ordem social desejável. Neste trabalho, são considerados os seguintes tipos de normas: (i) *obrigação* – os agentes devem realizar um resultado específico; (ii) *permissão* – os agentes podem agir de maneira particular e (iii) *proibição* – os agentes não devem agir de maneira específica. Todas essas características significam encorajar o cumprimento das normas através de recompensas e desestimular a violação das normas apontando as punições. Uma vez que o agente de software decide que sua prioridade é a satisfação de seus próprios desejos e metas, cada agente deve avaliar os efeitos associados ao cumprimento de uma ou mais normas antes de escolher qual deve ser cumprida. O mesmo se aplica quando os agentes decidem violar uma norma. Este trabalho apresenta uma estrutura para o desenvolvimento de SMA que fornecem mecanismos de suporte para a tomada de decisões do agente, usando o raciocínio prático baseado em normas. A aplicabilidade e validação desta abordagem é demonstrada ao aplica-la à um cenário de interseção de tráfego.

Palavras-chave: Agente BDI, BDI4JADE Framework, Sistema Multiagente, Agentes Normativos.

In charge of publications:

Rosane Teles Lins Castilho
Assessoria de Biblioteca, Documentação e Informação
PUC-Rio Departamento de Informática
Rua Marquês de São Vicente, 225 - Gávea
22451-900 Rio de Janeiro RJ Brasil
Tel. +55 21 3527-1516 Fax: +55 21 3527-1530
E-mail: bib-di@inf.puc-rio.br
Web site: http://bib-di.inf.puc-rio.br/techreports/

Table of Contents

1 Introduction

Multiagent systems are societies in which autonomous, heterogeneous and independently designed entities can work toward similar or different goals [1]. In order to deal with this autonomy and the diversity of interests among the different members, those open systems provide norms, which are mechanisms of social control to ensure a desirable social order [1]. Such mechanisms regulate the behavior of the agents by defining permission, obligation and prohibition [2]. Moreover, agents may be encouraged to fulfill a norm by obtaining rewards while being discouraged to violate it by receiving punishments [3]. Although norms are promising mechanisms to regulate an agent's behavior, the agent's autonomy might generate circumstances in which rather than fulfill the norm, the agent would prefer to violate it in order to reach a private goal that it considers to be more important. Within this context, new features were added to the BDI4JADE Framework [4] aiming to support normative reasoning, i.e., to build agents that are able to deal with desires and norms.

The original BDI4JADE Framework provides support only to the implementation of BDI agents and not the implementation of mechanisms that support normative functions. By using the proposed new features, it is possible to build BDI agents that are able to check if a norm should be adopted, or not. In addition, these new features evaluate the agent's desires and the effects of the fulfillment, or violation, of the norm. Lastly, it is possible to detect and solve conflicts among norms, and select desires and plans according to the agent's choice, i.e., whether the agent decides to fulfill a norm or not. The architectural support of this approach is provided by the NBDI (norm–belief–desire–intention) architecture [5], which extends the BDI (belief–desire–intention) architecture [6] by including norms-related functions to support normative reasoning. A traffic intersection scenario as well as the issues related to norms adoption, evaluation, and compliance are used to show the applicability of the new features.

The document is structured as follows. Section II focuses on the norms' background; presents the BDI4JADE framework, and offers an overview of the NBDI architecture. Section III presents related work. Section IV presents the NBDI4JADE architecture and details its implementation. Section V presents a usage scenario about traffic intersection norms in Brazil. Finally, Section VI shows the paper's conclusion and future work.

2 Background

This section summarizes the basic notions that will be used throughout this document, which aims to present the basic concepts about norms and their use in multiagent systems. The BDI4JADE framework and the NBDI architecture, which contribute to this work, are also presented.

2.1 Norms and Normative Multiagent Systems

Norms are informal rules that are socially enforced and represent an expected behavior towards a specific situation [7]. In the context of multiagent systems, norms are mechanisms commonly accepted as efficient means capable of regulating agent behavior and represent the way in which agents understand the responsibilities of other agents [8], [1]. Thus, agents work believing that other agents will behave according to the settled norms. Norms, however, are mainly mechanisms that enable agents to demand that

other agents behave in a certain way [9]. In addition, norms define permission obligation, or prohibition regarding the agents' behavior. Norms may be kept in place for different periods of time, i.e., either while the agent remains in the society or only for a short period of time, until the social goal has been fulfilled [10].

According to Mahmoud *et al*. [11], the literature suggests three different kinds of norms for normative multiagent systems [12] such as: (i) *regulative norms* which specify the behavior of a system by using obligations, prohibitions, and permissions [12]; (ii) *constitutive norms* which, besides regulating their own behavior can also create new norms derived from other existing norms [13–15]; (iii) *procedural norms* which are addressed to the agents in the normative system in order to regulate the behavior [16].

The definition of the norms used in this work [3] is represented by the following properties: Addressees, Condition (for example, Activation, Expiration), Motivation (for example, Rewards, Punishments), Deontic Concept, and States. The description of each property is given below: (i) Addressee is used to specify the agents or roles responsible for norm compliance; (ii) Activation is the condition for the norm to become active; (iii) Expiration is the validity condition for the norm to become inactive; (iv) Rewards is used to represent the set of rewards to be given to the agent for norm compliance; (v) Punishments is the set of punishments to be given to the agent for violating a norm; (vi) Deontic Concept is used to indicate whether the norm establishes an obligation, a permission, or a prohibition, and (vii) State is used to describe the set of states or actions that are being regulated.

Normative systems are widely discussed as a mechanism to regulate software agents [17]. Such systems are a set of constraints on the agents' behavior. By imposing these constraints, the intention is to enforce a social behavior. Normative systems are an important issue associated with software compliance. Norms are important whenever non-compliance is accidental (e.g., a message fails and some participants are not informed about the regulations). Alternatively, non-compliance may be deliberately rational (e.g., a participant chooses to ignore the norms because it does not see them as being in its own best interests), or deliberately irrational [18]. Furthermore, norms are important because they help shed light on the interaction of autonomous agents with one another and on how to control agent access to autonomous components [19].

2.2 The BDI4JADE Framework

BDI4JADE [4] is a framework based on the Java language that gives support to the development and implementation of Belief, Desire and Intention (BDI) agents – one of the widely known architectures for designing and implementing cognitive agents – and its implementation is a layer on top of the JADE platform [20], which provides a robust infrastructure to implement agents but does not support the BDI architecture.

Other BDI platforms based on the Java language, such as Jason [21], JACK [22], Jadex [23], and the 3APL Platform [24] have their agents implemented by using new programming languages – AgentSpeak(L) [25], JACK Agent Language [26], a Domain-specific Language (DSL) written in XML, and 3APL [27], respectively. This was the motivation behind the creation of the BDI4JADE Framework.

2.3 The NBDI Architecture

The NBDI architecture [5] extends the BDI architecture [6] by including norms-related functions to support normative reasoning. Moreover, norms are considered a primary concept that influences the agent's decision while reasoning about its beliefs, desires and

intentions. The extension of the NBDI architecture added three new components: (i) **Belief + Norm Review Function**; (ii) **Norm Selection Function**, and (iii) **Norm Filter** (see Figure 1).

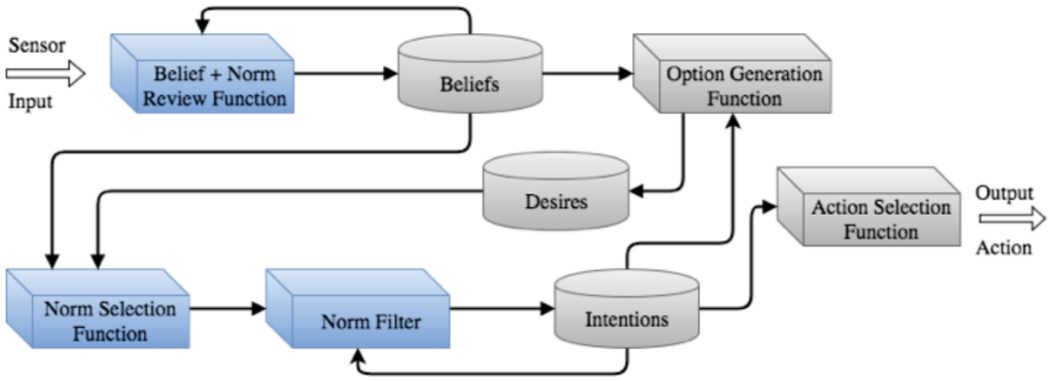

Fig. 1 BDI Architecture + NBDI Architecture

The *Belief + Norm Review Function* helps the agent to recognize its responsibilities towards other agents by adopting new norms that specify such responsibilities. In addition, it helps the agent to update the activated and adopted norms. This function consists of two tasks: (i) verifying the adopted norms and (ii) updating the norms. The first task checks if a new norm unifies with one of the norms already adopted, i.e., if the new norm already exists in the agent's belief base. This task further verifies if the agent is the addressee of the norm. Lastly, the first task updates the set of adopted norms in the agent's belief base if the new norm does not exist and the agent is the addressee of the norm. The second task updates the set of activated norms by evaluating the activation and expiration conditions and changing the status of the norm to "activated" or "deactivated".

The *Norm Selection Function* aims at selecting the norms that the agent has the intention to fulfill. To this end, this function first evaluates the status of the norms, the rewards, punishments and consequences and then, it detects and solves possible conflicts among the different norms that can be adopted.

Finally, the *Norm Filter* is responsible for discarding any intention that does not bring benefits to the agent, retaining intentions that are still expected and adopting new intentions. This function modifies the original BDI Filter Function, adding two additional steps: (i) selecting desires – this task selects the desires that will become intentions, taking into account the norms the agent wants to fulfill, and (ii) selecting plans – this task selects plans that are also influenced by the norms and will make the agents achieve their intentions.

3 Related Work

In the architecture of normative multiagent systems, the literature offers some research on normative systems. Following are some frameworks and their description.

BOID Normative Architecture: Broersen *et al.* proposed in this work, an architecture with an obligation component – the belief, obligation, intention, and desire (BOID) architecture. Such architecture adds an obligation component to the traditional BDI architecture and uses logical criteria to deal with the attitudes of the agent, with the changing environment and to resolve conflicts by according to the agent type. However, this approach does not address the danger of mandatory norms that may interpose the agent's autonomy [28].

BIO Normative Architecture: in this approach, Governatori and Rotolo proposed an architecture which considers the beliefs, intentions, and obligations as components. As well as the BOID architecture, BIO describes agents and their types in defeasible logic [29].

The OP-RND Normative Framework: this approach proposed a normative agent framework to regulate rules and norms effectively, the OP-RND framework. Their agents execute tasks based on pre-compiled tasks that considers their beliefs of the reward and penalty. Obligation and prohibition (OP) are rules imposed [30].

Boela *et al*. proposed an architecture of normative agents that uses deontic logic and is an extension of the work [31], specifying illegal behavior that an agent can carry out and its consequences [15].

4 NBDI4JADE – A Framework to Build Normative Agent

This section describes the main concepts of the proposed NBDI4JADE framework, providing an overview and discussing the different components that were changed, or added to the BDI4JADE framework, in order to allow NBDI4JADE to handle normative agents and to follow the concepts of the NBDI architecture. Furthermore, this section presents the NBDI4JADE class diagram and highlights details about its kernel (frozen-spots) and flexible points (hot-spots) [32].

4.1 The NBDI4JADE Framework

The NBDI4JADE framework supports the creation of simulations that show the impact of norms in multiagent systems. As such, NBDI4JADE enables the implementation of normative agents, allowing it to build complex multiagent systems and high-level abstraction.

To ensure a high-level abstraction, the NBDI4JADE framework was designed as a layer on top of other existing technologies, as shown in Figure 2. NBDI4JADE was built as an extension of the BDI4JADE framework, which is a BDI framework but does not support the norms concept. The design of NBDI4JADE considered the NBDI architecture, which presents, conceptually, the extension points and the changes needed in the BDI architecture in order to support normative reasoning agents. BDI4JADE, in turn, is a layer on top of the JADE framework, which provides a robust infra-structure to implement agents, but does not follow the BDI architecture.

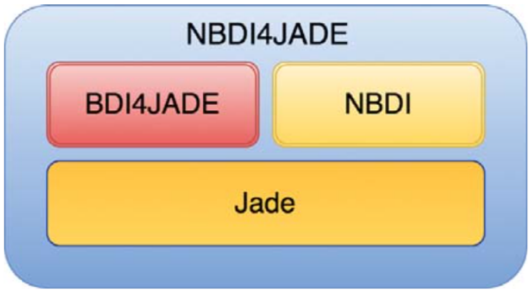

Fig. 2 The NBDI4JADE Architecture

4.2 Details of the NBDI4JADE Framework

The implementation of the NBDI4JADE framework aims at supporting the development of BDI agents capable of reasoning about their beliefs, desires and intentions, taking

norms into consideration. As such, the original components used in the reasoning cycle of the BDI4JADE agent, which is based on the BDI-interpreter algorithm presented in [6] were modified. The reasoning cycle is implemented in six major steps and each step is considered a component.

1. *Revising beliefs*: the first step consists of revising the agent's beliefs. This component was modified to enable agents to recognize their responsibilities towards other agents by adopting norms.

2. *Removing finished goals*: this step consists of removing goals that might have been "finished", i.e. the goals (i) may have been achieved, (ii) are no longer desired, or (iii) are considered unachievable. (This component was not modified.)

3. *Generating options*: in this step, are determined the goals (desires) that are available to the agent. This step is responsible for generate new desired goals; establish goals that are no longer desired, or preserve those goals that are still desired. This component was changed to generate options that take into account the norms of the environment.

4. *Removing dropped goals*: when a goal, or set of goals, is no longer considered desirable in the previous step, it is removed from the agent's set of goals and the observers are notified about this occurrence (This component was not modified.)

5. *Deliberating goals*: in this step, the current agent goals are partitioned into two subsets (i) goals to be achieved (intentions) and (ii) goals that are not achieved. The latter will remain an agent's desire, but the agent is not committed to achieve it at the moment. (This component was changed to consider the norms in the agent's belief base and to select plans that take the norms into account.)

6. *Updating goals status*: based on the partition performed in the previous step, the status of the goals is updated. Selected goals are updated to the "trying to achieve" status, and unselected goals are updated to the "waiting" status. When a goal has the "trying to achieve" status, the agent will select plans in order to achieve that goal. (This component was not modified.)

Figures 3–5 were designed using UML to demonstrate the changes that have been made to extend the BDI4JADE framework to deal with the norms concept. The red color indicates classes that already existed in BDI4JADE and were modified. The blue color indicates the new classes that were added to represent the norms concept. The gray color is that of those classes that did not suffer any changes in the deliberative process.

Figure 3 shows the new *DefaultBeliefNormRevisionStrategy* class that extends the *DefaultBeliefRevisionStrategy* class and adds the *reviewNorms* method. In addition, an interface to manage the DefaultBeliefNormRevisionStrategy class as well as two new classes to deal with the norms concept were added: (i) the *Norm* class, representing the structure of the basic concepts of a norm and (ii) the *NormBase* class, representing the set of norms of the system with their respective methods of manipulation.

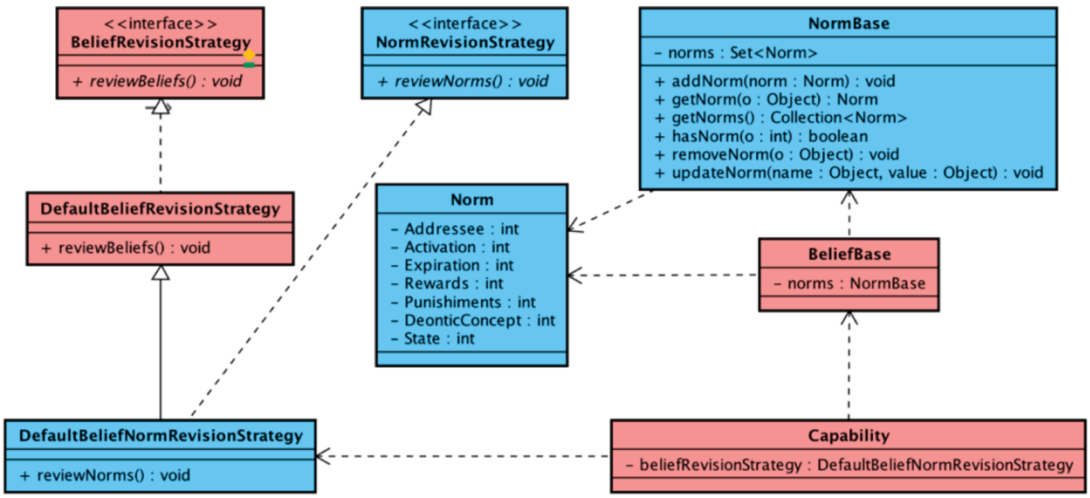

Fig. 3 Revision Norms and Beliefs Function

Figure 4 shows the changes in the deliberative goals function of the agent's reasoning cycle. As such, the agent's deliberative process considers the adoption of norms regarding its actions. The main changes occurred in the *DefaultAgentDeliberationFunction* class, which has received new methods to select goals, plans and filters that will consider the use of norms and their priorities.

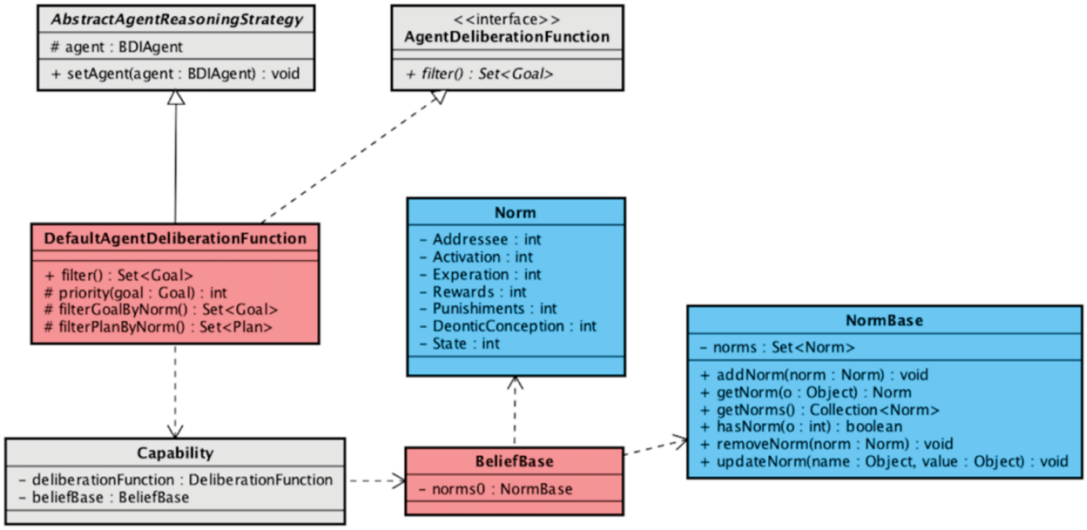

Fig. 4 Deliberative Goals Function

Figure 5 represents the intentional generation Function and the agent's plan to achieve its goals. The change in the *BeliefBase* class propagates to the *Capability* class and to the options generation class and agent selection plans. This change was a reflex of the new classes – *Norm* and *NormBase* – and it adds norms into the agent's reasoning cycle. As a result, the agent can decide whether to fulfill the norms or not by taking into account the norm's punishments and rewards. The goals generation and the plans selection functions take into account the concept of norm, which does not restrict the agent's autonomy. Therefore, the agent is now able to reason about the norms addressed to it. Such process is important when we consider normative conflicts.

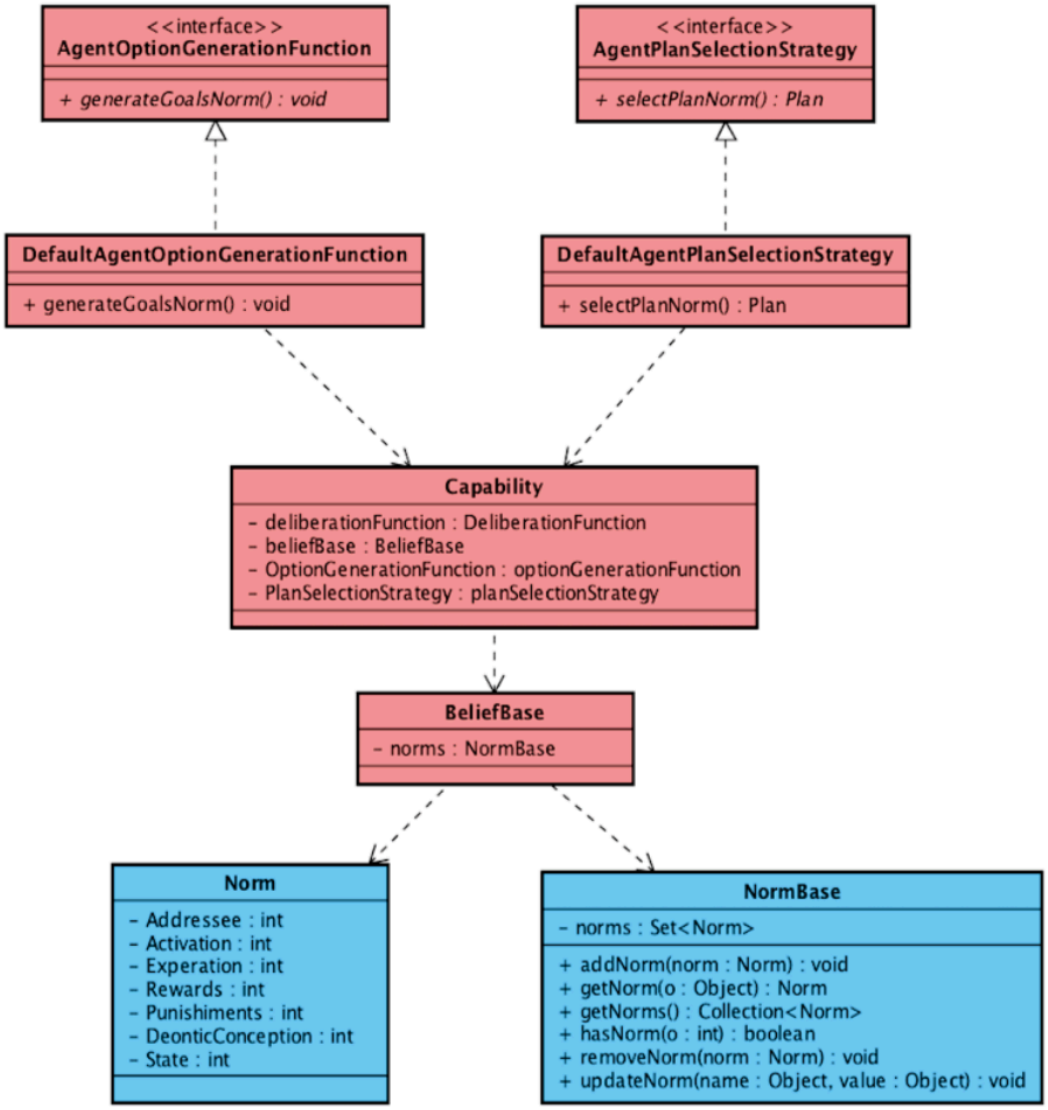

Fig. 5 Plan Generation Function

4.3 Hot-Spots and Frozen-Spots

Frameworks are generators of applications that are directly related to a specific domain [33]. This work proposes a framework whose domain is the development of normative agents.

Since frameworks are designed to generate complete applications, there must be flexible points that are customized to solve a particular problem. The initial proposed flexible point is restricted to the strategy used to deal with norms. According to their goals, agents can adopt a pressured, a rebellious or a social strategy in their decision-making process.

Some features of the framework are present in all applications in the domain. These immutable points constitute the core of a framework and are called fixed points (frozen-spots). The core is unchangeable and is also an ever-present part of every domain instance. However, there are also flexible parts in a framework providing extensible points (hot-spots) that are customized by developers. The hot-spots specifically defined by NBDI4JADE are:

- *DefaultBeliefNormRevisionStrategy*: it invokes the *NormRevisionStrategy.reviewNorms()* method for the norms base of all agents;
- *DefaultAgentOptionGenerationFunction*: it returns the current set of goals but takes into consideration the norms in place in the environment;
- *DefaultAgentDeliberationFunction*: it returns the whole set of goal, i.e., all goals will go to a "trying to achieve" state without violating the norms;
- *DefaultAgentPlanSelectionStrategy*: it returns null if the set of plans is empty, and the first plan retrieved from the set, otherwise, always respecting the norms imposed on the agent;

NBDI4JADE provides a default implementation for each one of these strategies and the hot-spots of BDI4JADE are maintained.

The frozen-spots of NBDI4JADE are:

- *NormBase*: it is the class that carries the methods for norm manipulation, i.e., it manages the environment's existing and active norms;

All fixed points (frozen-spots) of the BDI4JADE were maintained.

5 Usage Scenario: Traffic Intersection Norm in Brazil

The number of cars is continuously growing in Brazil. The large increase in the Brazilian fleet brought the number of cars to one car for every 4.4 inhabitants, i.e., it is estimated that there are approximately 45.4 million private vehicles in Brazil. Ten years ago, the proportion was 7.4 inhabitants per vehicle [9]. With the increase in the number of vehicles on the streets and the arrival of autonomous cars, the need arose to create systems capable of assisting both traffic experts as well as autonomous driver agents to better deal with unexpected situations in day-to-day traffic. The right of way rules at traffic intersections, for example, are difficult to follow at uncontrolled intersections, i.e., intersections without signs. Therefore, there are serious consequences when those rules are violated. An intersection is a junction where two or more roads meet, or cross.

According to data from the Brazilian Federal Highway Patrol [9], the main causes of fatal accidents in 2016 were, among others: lack of attention (30.8%); high speed (21.9%); alcohol consumption (15.6%); disregard for signs (10%); reckless overtaking (9.3%); and sleep (6.7%). In addition, 60% of these car accidents occurred at uncontrolled intersection. According to the Brazilian Transit Code (BTC),

Article 29, the right of way rules for vehicles arriving at an uncontrolled intersection are: (i) $\boldsymbol{Norm_1}$: vehicles moving on main thoroughfares have the preference; (ii) $\boldsymbol{Norm_2}$: in the case of a traffic circle, the ones circulating around it have the preference, and (iii) $\boldsymbol{Norm_3}$: in all other cases, vehicles coming from the right have the preference. In addition, **Article 38**, states that before making a right or left turn, or merging into traffic, the driver must, as per its *Sole paragraph*, yield to oncoming pedestrians, cyclists and vehicles, always respecting the norms of preference described in **article 29**.

The NBDI4JADE framework can simulate and assist in the planning of risk situations at uncontrolled intersections. For example, in order to avoid accidents, a simulation can be used to study the different strategies that can be adopted by normative autonomous car agents.

5.1 Overview

The simulation consists of autonomous cars, highways, traffic circles, and traffic intersections, respectively, as shown in Figure 6. The goal of the autonomous cars is to arrive at their destination without accidents. To achieve this goal, the autonomous car agent must be restricted by norms, but due to its autonomy, the agent may decide whether to fulfill these norms or not. Such simulations are, in fact, normative multiagent systems that receive data with the following information: (i) different types of traffic intersections, (ii) autonomous car agents, (iii) norms to be followed by the autonomous car agents, and (iv) different traffic scenarios. Simulations allow autonomous cars to find different solutions to prevent accidents at intersections.

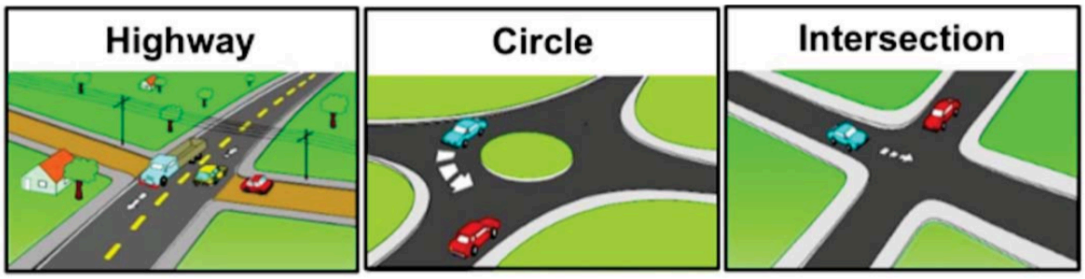

Fig. 6 Brazilian rules of preference at intersections

To deal with these scenarios and understand the norms applied to each scenario, the autonomous car agents have: (i) a set of goals that is connected directly to their individual satisfaction; (ii) a knowledge base collected by the simulation environment to help characterize traffic risk; and (iii) a set of strategies used to deal with the norms.

Figure 7 presents the following scenario: three cars arrive at an intersection at the same time. The agents' goals are: (i) The **PINK** autonomous car wants to proceed on street 1 and will have to cross street 2 in order to do so; (ii) The **YELLOW** autonomous car wants to proceed on street 2 and will have to cross street 1, and (iii) The **RED** autonomous car is on street 1 and wants to turn left onto street 2. However, there are no traffic signs and the agents need to be able to make decisions to avoid collision among the cars, taking into account the Brazilian traffic rules.

Fig. 7 Traffic Intersection rules in Brazil

As previously mentioned, **articles 29** and **38** of the BTC deal with the right of way rules at intersections. However, neither $Norm_1$ nor $Norm_2$ of **article 29** can be applied in this scenario. To solve this situation, the autonomous car agents need to decide whether they will fulfill or violate $Norm_3$ of **article 29**. This scenario considers that all autonomous car agents fulfilled all the norms. The agents' internal reasoning was built by using the NBDI4JADE framework and it is described below as if the agents had fulfilled $Norm_3$:

- The PINK autonomous car agent arrived at the intersection and stopped because the YELLOW car is on its right;
- The YELLOW autonomous car agent arrived at the intersection and stopped because the RED car is on its right;
- The RED autonomous car agent arrived at the intersection and there is no car on its right, therefore, the agent's reasoning cannot use **article 29**. To decide what to do, the agent needs to use **article 38**.

However, the Brazilian Transit Code (BTC) does not cover this situation, which creates an impasse. As such, we need to improve the agent's reasoning process in order to deal with this issue. Sometimes, the analysis of the BTC articles mentioned above will not be enough to allow the agent to make a decision. Consequently, it is necessary to consider different types of strategies that can be adopted by the agents to deal with the norms. For instance, in the scenario presented in Table I: (i) the PINK autonomous car agent adopts a *pressured strategy*, i.e., it fulfills the norms to achieve its individual goals, considering only the punishments that it will suffer; (ii) the YELLOW autonomous car agent adopts a *rebellious strategy*, i.e., it considers only their individual goals and violates all of the environment's norms, and (iii) the RED autonomous car agent adopts a *social strategy*, i.e., it complies with the norms and then verifies if it is possible to fulfill some of its individual goals. As a result, the PINK and RED autonomous car agents give the preference to the YELLOW autonomous car agent, which in turn accepts it because its rebellious strategy encourages this agent to go ahead.

TABLE I Strategies adopted by the autonomous car agents.

Strategies	PINK	YELLOW	RED
Pressured	X		
Rebellious		X	
Social			X

6 Conclusion and Future Work

This paper presents an initial architecture of an artificial agent that is able to make decisions by normative reasoning. The architecture is based on a NBDI architecture and BDI4JADE framework and was applied by modeling the traffic intersection rules in Brazil. The autonomous car agents make decisions about whether to continue or give the right of way by examining and reasoning about the norms of the environment and the presence, or absence, of any car near it.

As future work, this research aims to study how the concept of tests can be applied to verify normative systems. When agents start their decision making process, their decisions can lead to the violation of norms defined in the environment. An extension of the proposed architecture can be created to test normative MASs, allowing the extension

to check potential occurrences of such violations. *Can these violations modify the agent's goals? Can we track and record the agent's actions? Which types of tests should be developed to check the potential occurrences of violations? What happens to an environment when an agent violates a norm?* This research intends to answer these questions in future work. Last but not least, these tests will be applied in different usage scenarios, in order to evaluate norms violation based on the analysis of agents' behavior, thus understanding, to understand the fulfillment of the agent's internal goals.

References

[1] F. L. y López, "Social power and norms: Impact on agent behavior," Ph.D. dissertation, University of Southampton, 6 2003.

[2] N. Oren, M. Luck, and T. J. Norman, "Argumentation for normative reasoning," in Proc. Symp. Behaviour Regulation in Multi-Agent Systems, 2008, pp. 55–60.

[3] V. T. da Silva, "From the specification to the implementation of norms: an automatic approach to generate rules from norms to govern the behavior of agents," Autonomous Agents and Multi-Agent Systems, vol. 17, no. 1, pp. 113–155, 2008.

[4] I. Nunes, C. Lucena, and M. Luck, "Bdi4jade: a bdi layer on top of jade," ProMAS 2011, pp. 88–103, 2011.

[5] B. F. d. S. Neto, V. T. da Silva, and C. J. P. de Lucena, "Nbdi: An architecture for goal-oriented normative agents." in ICAART (1), 2011, pp. 116–125.

[6] A. S. Rao, M. P. Georgeff et al., "Bdi agents: From theory to practice." In ICMAS, vol. 95, 1995, pp. 312–319.

[7] A. Ahmad, "An agent-based framework incorporting rules, norms and emotions (oprnd-e)," Ph.D. dissertation, PhD Thesis, Universiti Tenaga Nasional, 2012.

[8] M. Alberti, A. Gomes, R. Gonçalves, J. Leite, and M. Slota, "Normative systems represented as hybrid knowledge bases," Computational Logic in Multi-Agent Systems, pp. 330–346, 2011.

[9] B. F. dos Santos Neto, V. T. Da Silva, and C. J. P. de Lucena, "Using jason to develop normative agents," in Brazilian Symposium on Artificial Intelligence. Springer, 2010, pp. 143–152.

[10] M. Luck, M. d'Inverno et al., "Constraining autonomy through norms," in Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 2. ACM, 2002, pp. 674–681.

[11] M. A. Mahmoud, M. S. Ahmad, M. Z. Mohd Yusoff, and A. Mustapha, "A review of norms and normative multiagent systems," The Scientific World Journal, vol. 2014, 2014.

[12] P. Caire, "A normative multi-agent systems approach to the use of conviviality for digital cities," Lecture Notes in Computer Science, vol. 4870, pp. 245–260, 2008.

[13] G. Boella and L. W. van der Torre, "Regulative and constitutive norms in normative multiagent systems." KR, vol. 4, pp. 255–265, 2004.

[14] R. Rubino, A. Omicini, and E. Denti, "Computational institutions for modelling norm-regulated mas: An approach based on coordination artifacts," In AAMAS Workshops. Springer, 2005, pp. 127–141.

[15] G. Boella and L. van der Torre, "An architecture of a normative system: counts-as conditionals, obligations and permissions," in Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems. ACM, 2006, pp. 229–231.

[16] G. Boella and L. van Der Torre, "Substantive and procedural norms in normative multiagent systems," Journal of Applied Logic, vol. 6, no. 2, pp. 152–171, 2008.

[17] T. Balke, C. da Costa Pereira, F. Dignum, E. Lorini, A. Rotolo, W. Vasconcelos, and S. Villata, "Norms in mas: definitions and related concepts," in Dagstuhl Follow-Ups, vol. 4. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2013.

[18] T. Ågotnes, W. van der Hoek, and M. Wooldridge, "Robust normative systems," in Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 2. International Foundation for Autonomous Agents and Multiagent Systems, 2008, pp. 747–754.

[19] O. Kafalı, N. Ajmeri, and M. P. Singh, "Kont: Computing tradeoffs in normative multiagent systems," in Proceedings of the 31st Conference on Artificial Intelligence (AAAI), To Appear, 2017.

[20] F. L. Bellifemine, G. Caire, and D. Greenwood, Developing multi-agent systems with JADE. John Wiley & Sons, 2007, vol. 7.

[21] R. H. Bordini, J. F. Hübner, and M. Wooldridge, Programming multi-agent systems in AgentSpeak using Jason. John Wiley & Sons, 2007, vol. 8.

[22] N. Howden, R. Rönnquist, A. Hodgson, and A. Lucas, "Jack intelligent agents-summary of an agent infrastructure," in 5th International conference on autonomous agents, 2001.

[23] L. Braubach, W. Lamersdorf, and A. Pokahr, "Jadex: Implementing a bdi-infrastructure for jade agents," 2003.

[24] 3APL - An Abstract Agent Programming Language, 2017 (accessed November 16, 2017), http://www.cs.uu.nl/3apl/.

[25] A. S. Rao,"Agentspeak(l): Bdi agents speak out in a logical computable language," in European Workshop on Modelling Autonomous Agents in a Multi-Agent World. Springer, 1996, pp. 42–55.

[26] M. Winikoff, "JackTM intelligent agents: an industrial strength platform," Multi-Agent Programming, pp. 175–193, 2005.

[27] M. Dastani, M. B. van Riemsdijk, F. Dignum, and J.-J. C. Meyer, "A programming language for cognitive agents goal directed 3apl," in International Workshop on Programming Multi-Agent Systems. Springer, 2003, pp. 111–130.

[28] J. Broersen, M. Dastani, and L. Van Der Torre, "Resolving conflicts between beliefs, obligations, intentions, and desires," in ECSQARU, vol. 1. Springer, 2001, pp. 568–579.

[29] G. Governatori and A. Rotolo, “Bio logical agents: Norms, beliefs, intentions in defeasible logic,” Autonomous Agents and Multi-Agent Systems, vol. 17, no. 1, pp. 36–69, 2008.

[30] A. Ahmad, M. Ahmed, M. Z. M. Yusof, M. S. Ahmad, and A. Mustapha, “Resolving conflicts between personal and normative goals in normative agent systems,” Journal of IT in Asia, vol. 4, no. 1, pp. 1–12, 2016.

[31] M. Alberti, M. Gavanelli, E. Lamma, P. Mello, P. Torroni, and G. Sartor, “Mapping deontic operators to abductive expectations,” Computational & Mathematical Organization Theory, vol. 12, no. 2-3, pp. 205–225, 2006.

[32] M. E. Fayad, D. C. Schmidt, and R. E. Johnson, Building application frameworks: object-oriented foundations of framework design. John Wiley & Sons, Inc., 1999.

[33] M. E. Markiewicz and C. J. de Lucena, “Object oriented framework development,” Crossroads, vol. 7, no. 4, pp. 3–9, 2001.