Extending Partial-Order Planning to Account for Norms in Agent Behavior

Tokimahery Ramarozaka, Jean-Pierre Müller, and Hasina Lalaina Rakotonirainy

Abstract Following a couple of models aiming to assess the effectiveness of norms in Madagascar on the MIMOSA platform, Müller et al., have noticed that the current architecture was not expressive enough to deal with all relevant norms, their different aspects, and how they interfere with the agent's behavior for such complex systems. In response, this paper proposes a new agent architecture and its dedicated language to enhance the expressiveness of norms in agent-based modeling. The architecture has to (1) identify all the applicable norms given a temporal, spatial, and social context, and (2) generate an agent behavior to account for these norms. We propose to extend automated planning and use a Model-Driven Engineering approach to build the abstract and concrete syntaxes of the language and its semantics. The resulting architecture will allow modelers to express a wider spectrum of norms and provides a normative decision tool that will ease further discussions and interpretations.

Keywords Automated planning · Agent-based modeling · Norms · Institutions · Model-driven engineering

1 Introduction

To answer the questions of norm effectiveness on renewable resource management, agent-based models (ABM) were explored with the MIRANA [1] and HINA [2] models, through the MIMOSA simulation platform [3]. However, both model's implementations were made ad-hoc, without any generic structure of norm, nor how they affect agent behavior, making them hard to (re)use and understand. Moreover, their lack of specification on a norm's spatial (where is a norm applicable?), temporal (when is it applicable?), and social context (to whom do they apply?) does not allow

T. Ramarozaka (✉) · H. L. Rakotonirainy
Informatique, Géomatique, Mathématiques et Décisions (IGMA), Andrainjato, University of Fianarantsoa, Fianarantsoa, Madagascar
e-mail: tokyramarozaka@gmail.com

J.-P. Müller
CIRAD—UMR SENS, Campus International de Baillarguet, 34398 Montpellier, France

F. Squazzoni (ed.), *Advances in Social Simulation*, Springer Proceedings in Complexity,
https://doi.org/10.1007/978-3-031-34920-1_11

modelers to express all relevant norms for such complex topics. For example, a norm that states that "it is forbidden to fish in winter around mangroves." might cause an agent to wait for the next season to fish, or to fish in non-mangrove areas; but how an agent reason and generate such behavior autonomously remains elusive.

While the literature on norms in ABM clearly offers a handful of normative agent architectures, such as BOID [4], NoA [5], or N-BDI [6], most of them focus on extending the BDI (Belief Desire Intention) architecture [7] with norms and does not provide the needed spatial, temporal and social context to be more expressive about norms. Recent works in [8] offer a mean to express these contexts and compute all applicable norms in that regard, but without describing how they are being applied in the agent decision-making process, i.e., in the agent architecture.

In response, this paper aims to propose a new agent architecture with its dedicated agent architecture to account for norms in agent behaviors while accounting for the spatial, temporal, and social context of norms. By using Model Driven Engineering (MDE) to build the language, our goal is to provide a tool to test the effectiveness of different norms on the agent's behavior and the whole socio-ecosystem.

Section 2 presents and justifies the methods we used to describe norms, and their influence on the agent's behavior. Section 3 then describes the results. Section 4 discusses how relevant they are for the issue at hand. Section 5 concludes by giving an overview of the current state of the project, and perspectives on future works.

2 Related Works

2.1 Norm Definition

In ABM, a norm is generally defined as a soft constraint on the agent's behavior [9, 10] to attain a desired behavior in time and space [11]. We represent this regulating mechanism through institutions [12], which is a set of ontology and norms. Each institution is endowed with a number of norms about the various social roles for the agents.

For instance, an institution could be a village where the *fisherman* role ought to fish only in authorized areas and have a license, while the *chief* role ought to provide enough food for his village. Norms can either tell what ought to be done under certain conditions, i.e. regulative norms; or, define that something counts as something else for a given institution, i.e. constitutive norms.

Constitutive norms introduce abstract classifications that some existing objects count as some concept or role within the institutions. They allow us to instantiate institutions as organizations by stating that an individual *John* counts as a fisherman, or that *Peter* counts as *chief* and that a certain place counts as a village.

Regulative norms are generally divided into three categories:

- Obligations: some action that the agent ought to do, or some fact that an agent ought to make true in a given situation;
- Prohibitions: some action that the agent ought not to do, or some fact that should not be true in a given situation;
- Permissions: some action or fact that is permitted in a certain situation(s) [13].

We chose the ADICO (Attribute Deontic aIm Condition Otherwise) formalism to express norms [14] as it covers most of the elements of norms in MAS, such as social roles describing the desired behaviors, rewards, and sanctions when applying or violating them [15], and more recently the notions of time and space formalized in [8, 16].

2.2 Model of Behavior

In classical AI, automated planning [17] is often used to generate agent behavior. It needs to define a planning problem with: some initial situation, a goal, and a set of possible actions, and outputs a sequence (or a partially ordered set) of actions called the *plan*, which allow the agent to reach its goal. The planning problem is then assimilated to a state-space search: a set of possible states is explored using operators to compute the next states until we find a solution state. Depending on the chosen approach, states and operators may encapsulate different concepts.

Since performance is not the main focus here, we chose Partial-Order Planning (POP) amongst other approaches for its flexibility: (1) states in POP are partial-order plans, they set constraints on the arrangement of the plan, rather than set a fixed sequence which would be harder to adapt when new or unexpected changes occur in the environment; (2) it is the most suited for agent planning as advocated in [18] because of the complexity of agent-based models. Each newly generated state (= plan) is examined for possible flaws like unachieved goals or conflicting actions. We compute the next state by solving one of those flaws, either: adding in new actions, arranging the partial-order of the plan, or constraining the variable bindings of its existing actions.

3 Our Contributions

3.1 Extending Planning Problems

A planning problem is defined by: some initial situation, a goal, and a set of possible actions, as summarized by Fig. 1. We represent actions in a deterministic way as in STRIPS [19], i.e. the consequences of the action are certain and occur simultaneously

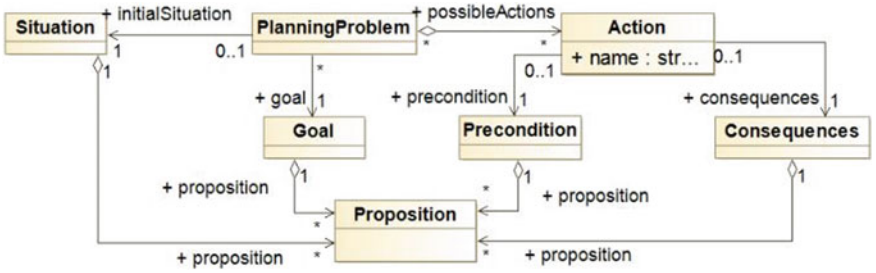

Fig. 1 Structure of a classic planning problem

when the action is executed. Dealing with conditional consequences is beyond the scope of this paper.

We propose to extend the planning problem (Fig. 1) with the concept of norms through organizations and institutions. Norms are applied to an agent through its roles within organizations, which are instances of one or more institutions.

For example, if the *fisherman* role is prohibited to fish in the *village* institution without a license, and the agent *Jon* plays the role of a *fisherman* in some *village v1* (*Jon* counts as a *fisherman*, *v1* counts as a *village*), then all the norms which would apply to *fisherman* apply to *Jon*: he is prohibited from fishing in *v*1 without a license. Thus, to define a planning problem with norms, we must describe the organizations in which the agent plays a role(s), listing all applicable norms, as shown in Fig. 2.

Each norm has a deontic operator (obligation, prohibition, permission) and applicability conditions with consequences that describe which action or proposition is mandatory, prohibited or permitted.

3.2 Extending POP's Internal Structure

States in classical POP are defined as plans, which we improve by correcting flaws to obtain an executable plan achieving the goal (if any). Formally, Fig. 3 defines a plan as a tuple <*S, A, Cc, Tc, F*> where:

- *S* is a set of situations;
- *A* is a set of steps, i.e. instances of actions;
- *Cc* is a set of (non) codenotation constraints describing that some variable must be bound to some value or not;
- *Tc* is a set of temporal constraints that describes which situation/step must be before another, defining a partial order;
- F is a set of flaws, which can either be open conditions: preconditions of steps that need to be satisfied, or threats: two conflicting steps, as stated in [20].

To compute the next state, a flaw in the plan is chosen and resolved.

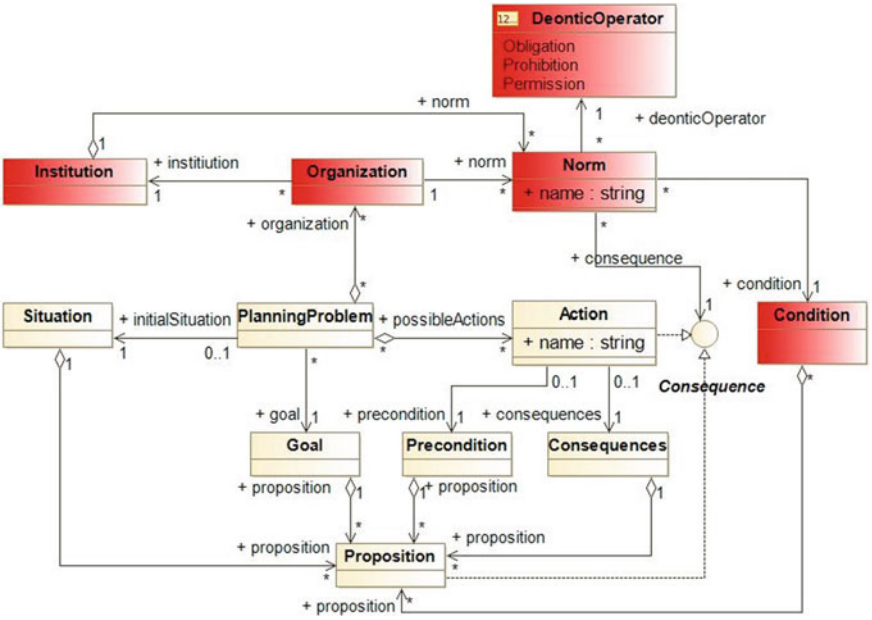

Fig. 2 Our extensions (highlighted in red) of a planning problem with norms

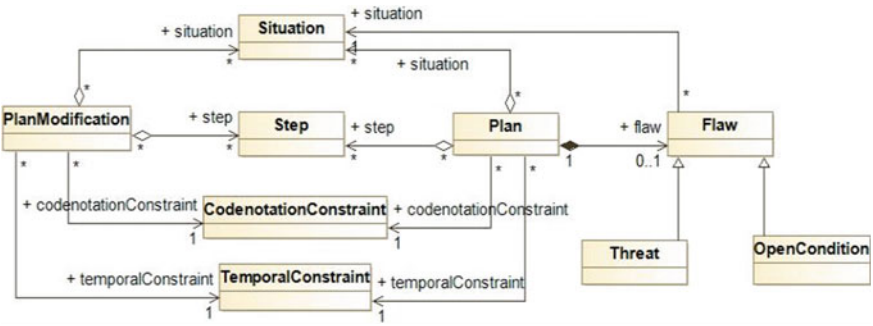

Fig. 3 Internal structure of a state in POP

Plan modification simply consists in adding situations, steps, (non) codenotation and/or temporal constraints to resolve a flaw. The planning stops when it has found an executable plan satisfying the goal. A plan is executable if, for each of its steps, all of its preconditions are necessarily true in its preceding situation, i.e. it does not contain open conditions; and its *Tc* and *Cc* do not contain contradictions. By introducing a dummy final step with the goal as its precondition, this executability condition is enough to find a solution plan (if any).

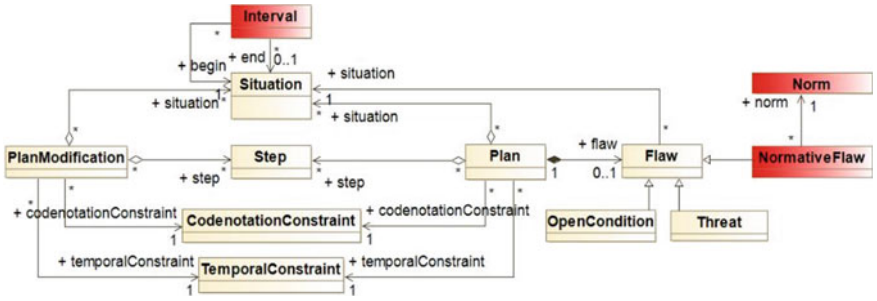

Fig. 4 Our extensions (highlighted in red) on the internal structure of a state in POP

Each state or partially ordered plan can be instantiated as a set of complete plans. A plan is complete if its *Tc* (temporal constraints) define a total order on its situations and steps, and its *Cc* (codenotation constraints) grounds all variables to a constant.

Throughout this paper, we shall refer to a proposition being «necessarily true», following Chapman's modal truth criterion [20] if that proposition is satisfied in any complete plan specified by a (partial) plan. We extend this existing structure with the notions of norm and interval, as highlighted in red in Fig. 4.

Since we introduce new flaws related to norms, we need to describe which plan modifications (operators) can resolve them.

3.3 Dealing with Norms

Our extensions to allow agents to deal with norms in the decision-making process rely on three key strategies: respect, violation, and circumvention. In the following section, we will take a look at how agents can respect or circumvent norms, and then see how they can violate them.

Dealing with obligations. Obligations are mandatory actions or propositions that need to be necessarily true in a certain situation of the plan. To consider them, we define a new flaw in POP.

Definition 1 (missing obligation). A missing obligation is a flaw such that the applicability conditions of an obligation are necessarily true in a situation *S*, and:

- the mandatory action *A* is not succeeding to *S*;
- or, the mandatory proposition *P* is not necessarily true in *S*

Any of the following new operators can be used to resolve them:

- *Promotion*: we take an existing instance of the mandatory action *A0* in the plan and we add the temporal constraint: $S < A0$ if the temporal constraints remain non-contradictory;

- *Adaptation*: add a new instance of the mandatory action *A0* and add the temporal constraint: $S < A0$;
- *Circumvention*: either, add a new step before *S* or add a temporal constraint that would make necessarily true the negation of one of the applicability conditions of the norm;

Dealing with prohibitions. Prohibitions are actions or propositions which are not allowed in a set of intervals, i.e. between any two situations: one where the prohibitions start to be applicable, and another situation where it is no longer prohibited.

Definition 2 (missing prohibition). A missing prohibition is a flaw such that the prohibition applicability conditions are necessarily true between the situations *Si* and *Sj*, and:

- The prohibited action is potentially between *Si* and *Sj*;
- or, the prohibited proposition *P* is satisfied in any situation s, such as $Si < s < Sj$.

We introduce the following operators to fix missing prohibitions:

- *Circumvention*: Adding a new step before the *Si*, which would make the negation of one of the applicability conditions necessarily true over the interval [*Si,Sj*];
- *Promotion* or *Demotion*: Add a new temporal constraint which would either (a) move the prohibited action before *Si* (*Promotion*); or (b) move the prohibited action after *Sj* (*Demotion*).

Dealing with permissions. Permissions are actions or propositions which are only allowed under certain conditions. A non-permitted action/proposition is automatically prohibited. To fix missing permissions, we generate for each state a list of prohibitions for all situations or intervals where the permission is not applicable and treat them according to the previous section.

Violating norms. If the agent fails to find a plan through normal planning, we propose that it starts to violate norms starting from the one endowed by the institution he values the least. In MIMOSA, each institution has a priority value that determines its importance. If the planner fails to find a solution when complying with all norms:

- The planner picks randomly a norm in the institution he values the least;
- It returns to the state before the associated normative flaw was resolved and re-plans by ignoring the related normative flaw;
- If the planner still fails, then another additional norm is violated, until a solution is found.

A basic use case of these extensions in renewable resource management would be a villager who ought to provide food for his family, but cannot hunt without a license

in his current area. This raises a flaw, as the villager's plan *to hunt* is in conflict with a prohibition. Using the aforementioned operators, he can either perform a:

- *Promotion* or *Demotion*: he will add steps to get a license before hunting;
- *Circumvention*: he will go elsewhere where he can hunt without a license;
- or else, a *Violation*: he'll consider illegal hunting, i.e. hunting without a license.

3.4 The Domain-Specific Language

To build a domain-specific language (DSL) implementing the previous extensions, we use a Model Driven Engineering (MDE) approach with three different artifacts:

(1) an abstract syntax describing the language's conceptual structure;
(2) a concrete syntax describing how to write the concepts;
(3) a semantic describing the meaning of a written sentence.

Abstract Syntax. The abstract syntax describes all the concepts of a DSL and their structure through a metamodel. Since we have already defined how we extend a planning problem with norms through Fig. 2, our DSL's abstract syntax applies these extensions to the abstract syntax of institutions (Fig. 5), and organizations (Fig. 6) by Müller & Raharivelo in [16], with an explicit representation of actions to build agent behavior. An institution is described by [16] as a set of words or concepts, typed by meta-concepts as seen in Fig. 5.

In summary, an institution is composed of the following elements:

- *meta* (for MetaConcept) enumerates all the meta-concepts;
- *concepts* are categorial concepts that can be denoted to sets of objects through *assertions*, they can also include notions of space or time;

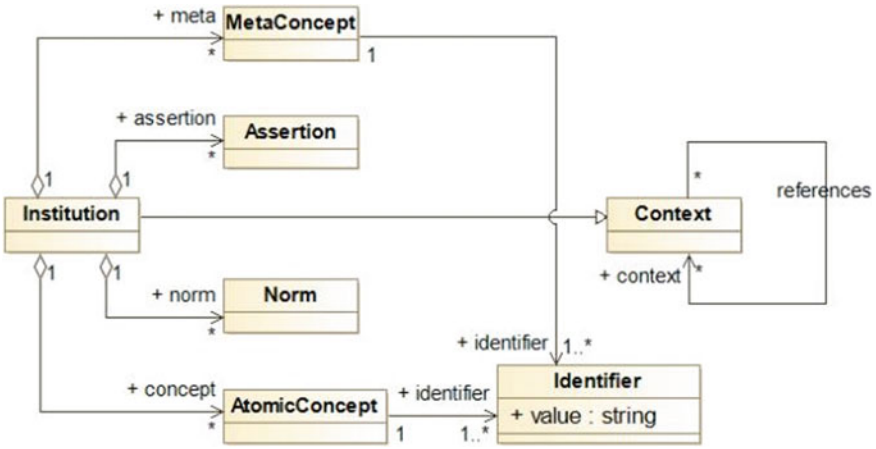

Fig. 5 Abstract syntax of an institution according to [16]

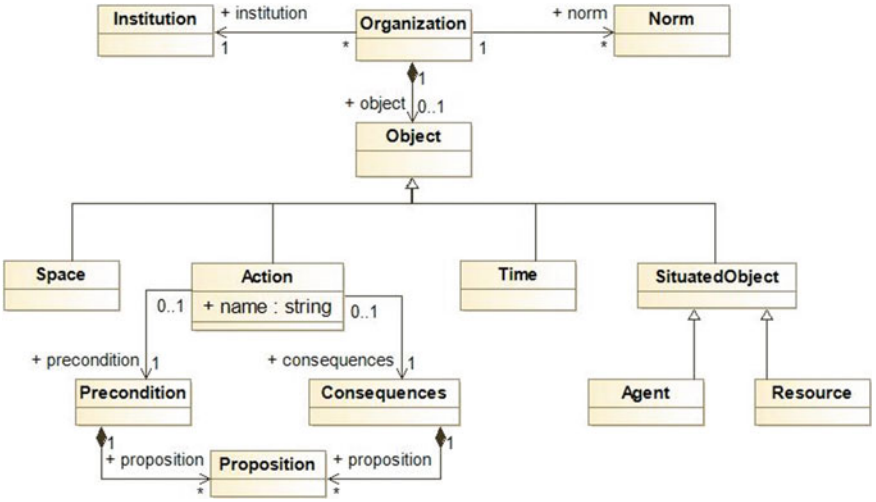

Fig. 6 Abstract syntax of an organization according to [16]

- *indiv* are individual concepts referring to individual objects;
- *norms* describe both regulative and constitutive norms alike.

By using *references*, concepts from other institutions can be imported or inherited. Since our main focus is to generate a sequence of actions, i.e. a plan, we add a STRIPS-like syntax [19] to describe actions by specifying its preconditions and consequences through a set of propositions. Preconditions are propositions that need to be necessarily true to execute the action, while consequences are composed of the propositions describing what the action adds and removes. In summary, the organization's abstract syntax from [16], is represented as in Fig. 6.

Lastly, as we describe the planning problem, we need to describe with the language the abstract syntax of an initial situation and a goal.

Both can be described through a set of propositions which are simple predicates from first-order logic, such as *hasFood()* or ¬*hasLicense()*, and can either be an affirmation or a negation (Fig. 7).

Concrete syntax. The concrete syntax depicts how we can use the concepts in the abstract syntax to build valid statements regarding the metamodel, either through some graphics or in some written form.

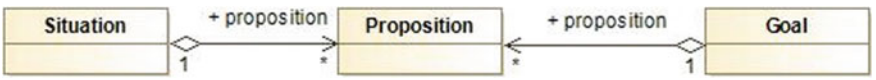

Fig. 7 Abstract syntax of a situation and a goal

Institutions. The concrete syntax of institutions, added to the definition of a planning problem if Fig. 2 is based upon the concrete syntax of institutions in [16] using the same annotated EBNF (Extended Backus-Naur Form) as follows.

<Institution> ::= 'institution' <institution_name> '{'

[<references>]
[<meta>]
[<concepts>]
[<indiv>]
[<norms>]
[<actions>]

'}'

Unlike the works in [16], our end goal is to produce plans that take norms into account, hence, the actions *component* is added to describe all possible actions which an institution brings about. We explicitly describe actions such as to *cuttingWood* by using a STRIPS representation with preconditions and consequences: the action requires to have an axe, and wood within the current area; executing the action adds the proposition "the agent now has wood", and removes the proposition "there is wood in the current area".

Organizations. While institutions can be perceived as virtual entities which define common norms and ontologies, organizations on the other hand are implementations of institutions that denote what will be associated with each concept and can define their own additional concepts if needed.

To describe organizations, we need to specify the institution it is implementing, and which agents or resources play which role through a set of constitutive norms. In terms of concrete syntax, Organizations remain the same as in [16], with the addition of an explicit action representation.

<Organization> ::='organization' <organization_name> from <institution_name>
'{'

[<meta>]
[<concepts>]
[<indiv>]
[<norms>]

'}'

Situations. Since an initial situation must be provided as a starting point for planning with the language, the following syntax can be used to describe the propositions in each situation.

<Situation> :: = 'situation' <situation_identifier> '{'

[<Propositions >].

'}'

Using the initial situation, more situations will be generated internally by the planner as it refines the plan with new actions and constraints to resolve its flaws.

4 Discussions

We proposed an approach to extend the POP algorithm with norms through new flaws which will allow agents to modify their plan according to applicable norms. The value of this work is to generate an appropriate agent behavior with regards to norms: either by applying, violating, or circumventing them.

By proposing strategies for an agent to circumvent norms in a spatio-temporal context, we can exhibit new behaviors such as: waiting for the right moment to fish, moving to another space where it is no longer prohibited to have a fishing license, acquiring a social status to be permitted to do something. This expressiveness makes it possible to understand the impact of norms on agent behavior.

The upcoming proof of concept will be done through a replica of the MIRANA model to demonstrate how effective the result is compared to the original ad-hoc implementation. Currently, we still need to take into account the spatial dimension of norms in planning, using spatial algebra like RCC8 (Region Connection Calculus), which has been applied to MAS in [16] to compute all applicable norms given a spatio-temporal context. The next step is to integrate them in our planner by (1) integrating the notions of time and space in situations or intervals of situations; and (2) further refining the proposed normative flaws by specifying which temporal or spatial context causes the normative flaw, and generate behavior accordingly.

While this approach allows us to take norms into account in the behavior generation process, some issues still have to be addressed: (1) the high complexity and performance of the algorithm, (2) the influence of sanctions and rewards on automated planning, and (3) the notion of quantity in planning.

5 Conclusion

In this paper, we tackle the issue with agent-based modeling when dealing with norms relative to complex issues such as renewable resource management, where agents are endowed with norms by the organizations in which they play a role in time and space. Since we need a better way to express and account for norms with their temporal, spatial, and social context, the aim of this paper is to propose an agent architecture and its dedicated language's abstract and concrete syntaxes which provides modelers with a tool to model such systems. In that regard, we used automated planning to allow agents to not only take into account or violate norms, but also to consider time and space in their decision-making. Norms are therefore structured in institutions,

and they are instantiated within organizations, which are instances of institutions. The resulting behavior model is built upon the POP paradigm, which allows the agent to exhibit new normative behaviors. The expected architecture, and its language will allow modelers to build one or multiple agents, set a number of norms, and see how they affect an agent's behavior.

References

1. Aubert, S., Müller, J.-P., Ralihalizara, J.: MIRANA: a socio-ecological model for assessing sustainability of community-based regulations. Int. Environ. Model. Softw. Soc. (iEMSs) 801–808 (2010)
2. Aubert, S., Müller, J.-P.: Incorporating institutions, norms and territories in a generic model to simulate the management of renewable resources. Artif. Intell. Law **21**(1), 47–78 (2013)
3. Müller, J.-P.: The mimosa generic modeling and simulation platform: the case of multi-agent systems. In: Proceedings of the 5th Workshop on Agent-Based Simulation, Lisbon, 2004. SCS. s.l.: s.n., 77–86. International Workshop on Agent-Based Simulation, 5, Lisbonne, Portugal, 3 Mai 2004/5 Mai 2004 (2004)
4. Broersen, J., Dastani, M., Hulstijn, J., van der Torre, L.: Goal generation in the BOID architecture. Cogn. Sci. Q. **2**(3–4), 428–447 (2002)
5. Kollingbaum, M.J., Norman, T.J.: NoA-a normative agent architecture. In: IJCAI, pp. 1465–1466 (2003)
6. dos Santos Neto, B.F., da Silva, V.T., de Lucena, CJ.: Developing goal-oriented normative agents: the NBDI architecture. In : International Conference on Agents and Artificial Intelligence, pp. 176–191. Springer, Berlin, Heidelberg (2011)
7. Rao, A.S., Georgeff, M.P.: Modeling rational agents within a BDI-architecture. In: Fikes, R., Sandewall, E. (eds.) Proceedings of Knowledge Representation and Reasoning (KR&R-91), pp. 473–484. Morgan Kaufmann Publishers, San Mateo, CA (1991)
8. Müller, J.P., Raharivelo, S.O.: Un méta-modèle pour représenter les normes dans un contexte multi-institutionnel territorialisé. Cépaduès (2017)
9. Cialdini, R.B., Trost, M.R.: Social influence: social norms, conformity and compliance. In: D.T. Gilbert, S.T. Fiske, G. Lindzey (eds.) The Handbook of Social Psychology, pp. 151–192. McGraw-Hill (1998)
10. Shoham, Y., & Tennenholtz, M.: On the synthesis of useful social laws for artificial agent societies (preliminary report). In: AAAI, pp. 276–281 (1992)
11. Dignum, V., Vázquez Salceda, J., Dignum, F.O.: Introducing Social Structure, Norms and Ontologies into Agent Organizations. Springer, pp 181–198 (2004)
12. Fornara, N., et al.: Modeling agent institutions. In: Agreement Technologies, pp. 277–307. Springer, Dordrecht (2013)
13. Ostrom, E.: Understanding Institutional Diversity. Princeton University Press (2009)
14. Crawford, S., Ostrom, E.: A grammar of institutions. Am. Polit. Sci. Rev. **89**(3), 582–600 (1995)
15. Interis, M.: On norms: a typology with discussion. Am. J. Econ. Sociol. **70**(2), 424–438 (2011)
16. Raharivelo, S.O., Müller, J.-P.: Un modèle de norme intégrant les conditions spatio-temporelles. In: JFSMA 2018. Systèmes Multi-Agents: Distribution et Décentralisation, pp 117–126. Cé-paduès (2018)
17. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach, 2nd edn. Prentice Hall/ Pearson Education (2003)
18. Kvarnström, J.: Planning for loosely coupled agents using partial order forward-chaining. In: Twenty-First International Conference on Automated Planning and Scheduling (2011)

19. Fikes, R.E., Nilsson, N.: STRIPS: a new approach to the application of theorem proving to problem solving. Artif. Intell. **2**(3–4), 189–208 (1971)
20. Chapman, D.: Planning for conjunctive goals. Artif. Intell. **32**(3), 333–377 (1987).