Instrumentalization of Norm-Regulated Transition System Situations

Magnus Hjelmblom[1,2]

[1] Faculty of Engineering and Sustainable Development, University of Gävle, Sweden
mbm@hig.se
[2] Department of Computer and Systems Sciences, Stockholm University, Sweden

Abstract. An approach to normative systems in the context of multi-agent systems (MAS) modeled as transition systems, in which actions are associated with transitions between different system states, is presented. The approach is based on relating the permission or prohibition of actions to the permission or prohibition of different types of state transitions with respect to some condition d on a number of agents $x_1, ..., x_\nu$ in a state. It introduces the notion of a norm-regulated transition system situation, which is intended to represent a single step in the run of a (norm-regulated) transition system. The normative framework uses an algebraic representation of conditional norms and is based on a systematic exploration of the possible types of state transitions with respect to $d(x_1, ..., x_\nu)$. A general-level Java/Prolog framework for norm-regulated transition system situations has been developed, and this implementation together with a simple example system is presented and discussed.
Keywords: transition system, multi-agent system, norm-regulated, norm-governed, normative system.

J. Filipe and A. Fred (Eds.): ICAART 2013, CCIS 449, pp. 8094, 2014.
DOI: 10.1007/978-3-662-44440-5_5

1 Introduction

Many dynamic systems, including multi-agent systems (MAS), may be modeled as transition systems, in which the actions of an agent are associated with transitions between different states of the system. There is a number of different approaches to normative systems in this context. The permission or prohibition of a specific action in a transition system is naturally connected to permissible or prohibited transitions between states of the system, and norms (sometimes referred to as 'social laws') may then be formulated as restrictions on states and state transitions.

This paper will introduce the notion of a *norm-regulated transition system situation*, which is intended to represent a single step in the run of a (norm-regulated) transition system. The permission or prohibition of actions in this framework is related to the permission or prohibition of different types of state transitions with respect to some condition d on a number of agents $x_1, ..., x_\nu$ in a state. The framework uses an algebraic representation of conditional norms, based on the representation used in the norm-regulated DALMAS architecture (see Previous Work, Sect. 1.2). The novel feature presented here is primarily an extension to the DALMAS's normative framework, based on a systematic exploration of the possible types of state transitions with respect to $d(x_1, ..., x_\nu)$. A norm-regulated transition system situation is easily instrumentalized into a general-level Prolog module that can be used to implement a wide range of specific norm-regulated dynamic systems.

Important norm-related issues such as enforcement of norms, norm change and consistency of normative systems are beyond the scope of this paper; however, the approach presented here is general in nature, and may be combined with many different approaches to, e.g., norm enforcement. The term 'agent' will be frequently used for some sort of 'acting entity' within a dynamic system, but no special assumptions are made about for example autonomy, reasoning capability, architecture, and so on.

1.1 Transition System Situations

A labelled transition system (LTS) is usually defined (see for example [4, p. 174]) as an ordered 3-tuple $\langle S, E, R\rangle$ where S is a non-empty set of *states*; E is a set of transition *labels*, often called *events*; and $R \subseteq S \times E \times S$ is a non-empty set of labelled *transitions*. If (s, ε, s') is a transition, s is the initial state and s' is the resulting state of ε. An event ε is *executable* in a state s if there is a transition $(s, \varepsilon, s') \in R$, and *non-deterministic* if there are transitions $(s, \varepsilon, s') \in R$ and $(s, \varepsilon, s^*) \in R$ with $s' \neq s^*$. A *path* (or *run*) of length m ($m \geq 0$) of a labelled transition system is a sequence $s_0\varepsilon_0 s_1 \cdots s_{m-1}\varepsilon_{m-1}s_m$ such that, for $i \in \{1, .., m\}$, $(s_{i-1}, \varepsilon_{i-1}, s_i) \in R$.

In the following, we restrict our attention to transition systems in which all events are deterministic. This means that, for each state s, the labels associated with the outgoing transitions from s are distinct. Furthermore, we assume that a ν-ary condition d is true or false on ν agents $x_1, ..., x_\nu \in \Omega$ in s, where Ω

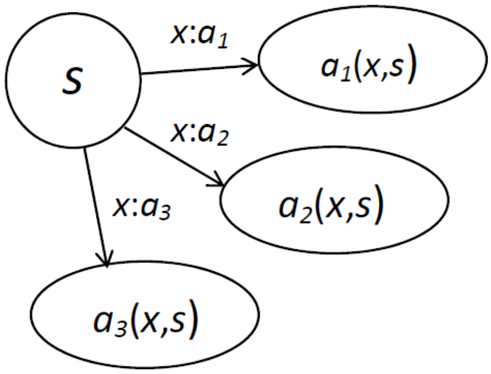

Fig. 1. A state diagram for a transition system situation with three events.

is a set of agents associated with s; this will be written $d(x_1, ..., x_\nu; s)$. In the special case when the sequence of agents is empty, i.e. $\nu = 0$, d represents a proposition which is true or false in s. Let us now focus on an arbitrary state in a deterministic LTS, with the added requirement that each event ε represents an action a performed by a single agent x. This is written $\varepsilon = x{:}a$, referring to both to the moving agent x and an action a. The term *transition system situation* will be used for an ordered 5-tuple $\mathbf{S} = \langle x, s, A, \Omega, S\rangle$ characterized by a set of states S, a state s, an agent-set $\Omega = \{x_1, ..., x_n\}$, the acting ('moving') agent x, and an action-set $A = \{a_1, ..., a_m\}$. In this setting, a may be regarded as a function such that $a(x, s) = s^+$ means that s^+ is the resulting state when x performs act a in state s.[3] In the following, the abbreviation s^+ will be used for $a(x, s)$ when there is no need for an explicit reference to the action a and the acting agent x.

As indicated by Fig. 1, a transition system situation is intended to represent, for example, a 'snapshot' of a labelled transition system in which each transition is deterministic and represents the action of a single agent. In this case, s represents an arbitrarily chosen state in the LTS, and S is the set of states reachable from s by all transitions $x{:}a$, $a \in A$. At the same time, a transition system situation is designed to be general enough to also represent a step in a run of other kinds of dynamic systems, including systems modeled by finite automata (see for example [11]) or Petri nets, and deterministic Dalmases.

1.2 Related Work

This section will give a brief overview of different approaches to the design of normative systems and the formulation of norms. A common feature of many approaches is the idea to partition states and (possibly) transitions into two categories, for example 'permitted' and 'non-permitted'. This may be accomplished

[3] Note that no special assumptions are made regarding whether or not s_0 is an element of S, i.e. whether or not the action a may lead back to s_0.

with the use of if-then-else rules or constraints on the states and/or the transitions between states. The Ballroom system in [5] and the anticipatory system for plot development guidance in [11] both serve as examples of this approach. Some approaches are purely algebraic or based on modal logics, for example temporal or deontic logic. The DALMAS architecture (see Previous Work below) for norm-regulated MAS is based on an algebraic approach to the representation of normative systems. Dynamic deontic logic [20] and Dynamic logic of permission [19] are two well-known examples of the modal logic approach. Other examples are the combination of temporalised agency and temporalised normative positions [6], in the setting of Defeasible Logic, and Input/Output Logic by Makinson and van der Torre (see for example [18]). Vázquez-Salceda et al. use a language consisting of deontic concepts which can be conditional and can include temporal operators. They characterize norms by whether they refer to states (i.e., norms concerning that an agent sees to it that some condition holds) or actions (i.e., norms concerning an agent performing a specific action), whether they are conditional, whether they include a deadline, or whether they are norms concerning other norms. [26] $n\mathcal{C}+$, an extension of the action language $\mathcal{C}+$, is employed within the context of 'coloured agent-stranded transition systems' [4] to formulate two kinds of norms: *state permission laws* and *action permission laws*. A state permission law states that certain (types of) states are permissible or prohibited, while an action permission law states that specific (types of) transitions are permissible or prohibited in certain states. By picking out the component ('strand') corresponding to an individual agent's contribution to an event, different categories of non-compliant behaviour ('sub-standard' resp. 'unavoidably non-compliant' behaviour) can be distinguished. Cliffe et al. use Answer Set Programming (ASP) for representing institutional norms, as part of the representation and analysis of specifications of agent-based institutions. [1, 2] In Deontic Petri nets, and variants thereof such as Organizational Petri nets, varying degrees of 'ideal' or 'sub-ideal' (more or less 'allowed' or 'preferred') behaviour is modeled by preference orderings on executions of Petri nets; see for example [23, 3].

Previous Work: The Dalmas Architecture DALMAS [22] is an abstract architecture for a class of (norm-regulated) multi-agent systems. A deterministic DALMAS is a simple multi-agent system in which the actions of an agent are connected to transitions between system states. In a deterministic DALMAS the agents take turns to act; only one agent at a time may perform an action. Therefore, each individual step in a run of the system may be represented by a transition system situation.

A DALMAS is formally described by an ordered 9-tuple, where the arguments are various sets, operators and functions which give the specific DALMAS its unique features. Of particular interest is the deontic structure-operator, which for each situation of the system determines an agent's *deontic structure* (i.e., the set of permissible acts) on the feasible acts in the current situation, and the preference structure-operator, which for each situation determines the *preference* *structure* on the permissible acts. In a *norm-regulated simple deterministic* DALMAS, the deontic structure consists of all acts that are not explicitly prohibited by a normative system; thereby employing what is often referred to as 'negative permission'. The preference structure consists of the most preferable (according to the agent's *utility function*) of the acts in the deontic structure. In other words, a DALMAS agent's behaviour is regulated by the combination of a normative system and a utility function. The normative system consists of conditional norms using the Kanger-Lindahl theory of normative positions, expressed in an algebraic notation for norms. See for example [12, 14, 21] for an introduction. A general-level Java/Prolog implementation of the DALMAS architecture has been developed, to facilitate the implementation of specific systems. The **Colour & Form** system, the **Waste-collector** system and the **Forest Cleaner** system are three specific systems that have been implemented using this framework. The reader is referred to [22, 7, 10, 8] for a description of these systems and their implementations.

2 Normative Systems and Types of State Transitions

In this section, which is a slight reformulation of Sect. 2 in [9], we consider the transition from a state s to a following state s^+, and focus on the condition $d(x_1, ..., x_\nu)$. To facilitate reading, X_ν will be used as an abbreviation for the argument sequence $x_1, ..., x_\nu$. With regard to $d(X_\nu)$, there are four possible alternatives for the transition from s to s^+, since in s as well as in s^+, $d(X_\nu)$ or $(d^\neg)(X_\nu)$ could hold[4]:

I. $d(X_\nu; s)$ and $d(X_\nu; s^+)$
II. $\neg d(X_\nu; s)$ and $d(X_\nu; s^+)$
III. $d(X_\nu; s)$ and $\neg d(X_\nu; s^+)$
IV. $\neg d(X_\nu; s)$ and $\neg d(X_\nu; s^+)$

Each alternative represents a basic type of transition with regard to the state of affairs $d(X_\nu)$; we say that $\{\mathrm{I}, \mathrm{II}, \mathrm{III}, \mathrm{IV}\}$ is the set of *basic transition types* with regard to $d(X_\nu)$. In the vein of [24], I could be written $0{:}d(X_\nu) \wedge 1{:}d(X_\nu)$, II could be written $0{:}\neg d(X_\nu) \wedge 1{:}d(X_\nu)$, and similarly for III and IV.

Let the *situation* $\langle x, s\rangle$ be characterized by the moving agent x and the state s in a transition system situation $\mathbf{S}$. We now wish to be able to determine the transition type for the transition represented by action a performed by agent $x_{\nu+1}$ in $\langle x, s\rangle$. (The point of the separation between $x_{\nu+1}$ and the moving agent x is to allow for systems in which normative conditions may apply to other agents than the 'moving' agent, e.g. agents wishing to perform some sort of 'reaction' or

[4] We can form negations $(d^\neg)$ of conditions in the following way: $(d^\neg)(X_\nu)$ iff $\neg d(X_\nu)$. In the following, the latter notation will be used to facilitate the presentation. Note that conjunctions $(c \wedge d)$ and disjunctions $(c \vee d)$ may be formed in a similar way; hence, it is possible to construct Boolean algebras of conditions.

'punishment' act. In most simple systems, however, $x_{\nu+1}$ will be identified with x.) Therefore, we define a 'basic transition type operator' B^a_j, $j \in \{\mathrm{I}, \mathrm{II}, \mathrm{III}, \mathrm{IV}\}$, such that the $\nu + 1$-ary 'transition type condition' $B^a_j d(X_\nu, x_{\nu+1}; x, s)$ indicates whether or not, in the situation $\langle x, s\rangle$, the event $x_{\nu+1}{:}a$ (representing a being performed by $x_{\nu+1}$) has basic transition type j with regard to $d(X_\nu)$: For all ν-ary conditions d and for all agents $X_\nu, x_{\nu+1}$, all acts a and all situations $\langle x, s\rangle$,

1. $B^a_{\mathrm{I}} d(X_\nu, x_{\nu+1}; x, s)$ iff $[d(X_\nu; s) \wedge d(X_\nu; a(x_{\nu+1}, s)]$
2. $B^a_{\mathrm{II}} d(X_\nu, x_{\nu+1}; x, s)$ iff $[\neg d(X_\nu; s) \wedge d(X_\nu; a(x_{\nu+1}, s))]$
3. $B^a_{\mathrm{III}} d(X_\nu, x_{\nu+1}; x, s)$ iff $[d(X_\nu; s) \wedge \neg d(X_\nu; a(x_{\nu+1}, s))]$
4. $B^a_{\mathrm{IV}} d(X_\nu, x_{\nu+1}; x, s)$ iff $[\neg d(X_\nu; s) \wedge \neg d(X_\nu; a(x_{\nu+1}, s))]$

We note in passing the following symmetries:

– $B^a_{\mathrm{I}} d(X_\nu, x_{\nu+1}; x, s)$ iff $B^a_{\mathrm{IV}}(d^\neg)(X_\nu, x_{\nu+1}; x, s)$
– $B^a_{\mathrm{II}} d(X_\nu, x_{\nu+1}; x, s)$ iff $B^a_{\mathrm{III}}(d^\neg)(X_\nu, x_{\nu+1}; x, s)$

2.1 Prohibition of State Transition Types

$\{\mathrm{I}, \mathrm{II}, \mathrm{III}, \mathrm{IV}\}$ is the set of atoms of the boolean algebra generated by $d(X_\nu; s)$ and $d(X_\nu; s^+)$. This algebra has 16 elements, as shown in Table 1, where 'X' denotes that a basic transition type is one of the disjuncts of the element, while '-' denotes that it is not. I.e., each subset of $\{\mathrm{I}, \mathrm{II}, \mathrm{III}, \mathrm{IV}\}$ represents a combination (by disjunction) of basic transition types with regard to $d(X_\nu)$. For each element (i.e., for each row in the table) we obtain conditions on state transitions. E.g., row 5 represents the condition $\neg d(X_\nu; s) \wedge d(X_\nu; s^+)$, and row 8 represents the condition

$$(\neg d(X_\nu; s) \wedge d(X_\nu; s^+)) \vee (d(X_\nu; s) \wedge \neg d(X_\nu; s^+)) \vee (\neg d(X_\nu; s) \wedge \neg d(X_\nu; s^+))$$

which may be simplified to $\neg d(X_\nu; s) \vee \neg d(X_\nu; s^+)$.

The idea now is to formulate (conditional) norms whose normative consequents prohibit one or more basic transition types. A specific act a is taken to be prohibited for $x_{\nu+1}$ if, in a certain state s, the normative system contains a norm which prohibits the type of transition represented by $x_{\nu+1}{:}a$. For each transition type condition, i.e. for each row in Table 1, we may now stipulate that if the transition type condition holds of the transition $(s, x_{\nu+1}{:}a, a(x_{\nu+1}, s))$ then it is not permissible for $x_{\nu+1}$ to perform a in the situation $\langle x, s\rangle$. E.g., for row 5 we may stipulate that if $\neg d(X_\nu; s)$ and $d(X_\nu; a(x_{\nu+1}, s))$, then act a is not permissible for $x_{\nu+1}$ in $\langle x, s\rangle$, by defining a normative operator P_{II} such that for all ν-ary conditions d, all agents $X_\nu, x_{\nu+1} \in \Omega$, all actions $a \in A$, and all situations $\langle x, s\rangle$,

$$P_{\mathrm{II}} d(X_\nu, x_{\nu+1}; x, s) \text{ iff}$$
$$[\text{if } B^a_{\mathrm{II}} d(X_\nu, x_{\nu+1}; x, s), \text{ then } a \text{ is prohibited for } x_{\nu+1}].$$

Table 1. Possible Combinations of Basic Transition Types.

	(I)	(II)	(III)	(IV)	
1	-	-	-	-	-
2	-	-	-	X	$\neg d(X_\nu; s) \wedge \neg d(X_\nu; s^+)$
3	-	-	X	-	$d(X_\nu; s) \wedge \neg d(X_\nu; s^+)$
4	-	-	X	X	$\neg d(X_\nu; s^+)$
5	-	X	-	-	$\neg d(X_\nu; s) \wedge d(X_\nu; s^+)$
6	-	X	-	X	$\neg d(X_\nu; s)$
7	-	X	X	-	$\neg(d(X_\nu; s) \leftrightarrow d(X_\nu; s^+))$
8	-	X	X	X	$\neg d(X_\nu; s) \vee \neg d(X_\nu; a(x, s))$
9	X	-	-	-	$d(X_\nu; s) \wedge d(X_\nu; s^+)$
10	X	-	-	X	$d(X_\nu; s) \leftrightarrow d(X_\nu; s^+)$
11	X	-	X	-	$d(X_\nu; s)$
12	X	-	X	X	$d(X_\nu; s) \vee \neg d(X_\nu; s^+)$
13	X	X	-	-	$d(X_\nu; s^+)$
14	X	X	-	X	$\neg d(X_\nu; s) \vee d(X_\nu; s^+)$
15	X	X	X	-	$d(X_\nu; s) \vee d(X_\nu; s^+)$
16	X	X	X	X	$\top$

Similarly, for row 8 we may define $P_{\mathrm{II,III,IV}}$ such that for all ν-ary conditions d, all agents $X_\nu, x_{\nu+1} \in \Omega$, all actions $a \in A$, and all situations $\langle x, s\rangle$,

$$\begin{gathered} P_{\mathrm{II,III,IV}} d(X_\nu, x_{\nu+1}; x, s) \text{ iff} \\ [\text{if } B^a_{\mathrm{II}} d(X_\nu, x_{\nu+1}; x, s) \text{ or } B^a_{\mathrm{III}} d(X_\nu, x_{\nu+1}; x, s) \text{ or } B^a_{\mathrm{IV}} d(X_\nu, x_{\nu+1}; x, s), \\ \text{then } a \text{ is prohibited for } x_{\nu+1}]. \end{gathered}$$

A closer look at Table 1 reveals, however, that not all disjunctions of basic transition types can be meaningfully linked with a prohibition. As discussed in [9], norms based on the prohibition of elements containing $\mathrm{I} \vee \mathrm{III}$ or $\mathrm{II} \vee \mathrm{IV}$ are not meaningful. Table 2 contains the rows (slightly reordered) that represent meaningful normative conditions. It is convenient to define a 'transition type operator' C^a_j, $j \in \{2, 2', 4, 4', 5, 6, 6', 7\}$, for each of the rows in Table 2 (except the first, which expresses no restrictions at all)[5]:

For all ν-ary conditions d and for all agents $X_\nu, x_{\nu+1}$, all acts a and all situations $\langle x, s\rangle$,

1. $C^a_2 d(X_\nu, x_{\nu+1}; x, s)$ iff $B^a_{\mathrm{III}} d(X_\nu, x_{\nu+1}; x, s)$ iff $[d(X_\nu; s) \wedge \neg d(X_\nu; a(x_{\nu+1}, s)]$
2. $C^a_{2'} d(X_\nu, x_{\nu+1}; x, s)$ iff $B^a_{\mathrm{IV}} d(X_\nu, x_{\nu+1}; x, s)$ iff $[\neg d(X_\nu; s) \wedge \neg d(X_\nu; a(x_{\nu+1}, s))]$
3. $C^a_4 d(X_\nu, x_{\nu+1}; x, s)$ iff $B^a_{\mathrm{II}} d(X_\nu, x_{\nu+1}; x, s)$ iff $[\neg d(X_\nu; s) \wedge d(X_\nu; a(x_{\nu+1}, s))]$

[5] The numbering is based on the numbering used in [8].

4. $C^a_{4'}d(X_\nu, x_{\nu+1}; x, s)$ iff $B^a_{\mathrm{I}}d(X_\nu, x_{\nu+1}; x, s)$ iff $[d(X_\nu; s) \wedge d(X_\nu; a(x_{\nu+1}, s))]$
5. $C^a_5 d(X_\nu, x_{\nu+1}; x, s)$ iff $[B^a_{\mathrm{III}}d(X_\nu, x_{\nu+1}; x, s)$ or $B^a_{\mathrm{IV}}d(X_\nu, x_{\nu+1}; x, s)]$ iff $\neg d(X_\nu; a(x_{\nu+1}, s))$
6. $C^a_6 d(X_\nu, x_{\nu+1}; x, s)$ iff $[B^a_{\mathrm{II}}d(X_\nu, x_{\nu+1}; x, s)$ or $B^a_{\mathrm{III}}d(X_\nu, x_{\nu+1}; x, s)]$ iff $[\neg d(X_\nu; s) \wedge d(X_\nu; a(x_{\nu+1}, s))] \vee [d(X_\nu; s) \wedge \neg d(X_\nu; a(x_{\nu+1}, s))]$
7. $C^a_{6'} d(X_\nu, x_{\nu+1}; x, s)$ iff $[B^a_{\mathrm{I}}d(X_\nu, x_{\nu+1}; x, s)$ or $B^a_{\mathrm{IV}}d(X_\nu, x_{\nu+1}; x, s)]$ iff $[d(X_\nu; s) \wedge d(X_\nu; a(x_{\nu+1}, s))] \vee [\neg d(X_\nu; s) \wedge \neg d(X_\nu; a(x_{\nu+1}, s))]$
8. $C^a_7 d(X_\nu, x_{\nu+1}; x, s)$ iff $[B^a_{\mathrm{I}}d(X_\nu, x_{\nu+1}; x, s)$ or $B^a_{\mathrm{II}}d(X_\nu, x_{\nu+1}; x, s)]$ iff $d(X_\nu; a(x_{\nu+1}, s))$

The 'transition type condition' $C^a_j d(X_\nu, x_{\nu+1}; x, s)$ indicates whether or not, in situation $\langle x, s\rangle$, the event $x_{\nu+1}{:}a$ has any of the corresponding basic transition types with regard to $d(X_\nu)$. The following symmetries hold (cf. the observation in [22, p. 148]):

– $C^a_2 d(X_\nu, x_{\nu+1}; x, s)$ iff $C^a_4(d^\neg)(X_\nu, x_{\nu+1}; x, s)$
– $C^a_{2'} d(X_\nu, x_{\nu+1}; x, s)$ iff $C^a_{4'}(d^\neg)(X_\nu, x_{\nu+1}; x, s)$
– $C^a_5 d(X_\nu, x_{\nu+1}; x, s)$ iff $C^a_7(d^\neg)(X_\nu, x_{\nu+1}; x, s)$
– $C^a_6 d(X_\nu, x_{\nu+1}; x, s)$ iff $C^a_6(d^\neg)(X_\nu, x_{\nu+1}; x, s)$
– $C^a_{6'} d(X_\nu, x_{\nu+1}; x, s)$ iff $C^a_{6'}(d^\neg)(X_\nu, x_{\nu+1}; x, s)$

Next, we define a normative 'transition type prohibition operator' P_1 such that it imposes no restriction on the actions performed by $x_{\nu+1}$, and, for each C^a_j, a transition type prohibition operator P_j, $j \in \{2, 2', 4, 4', 5, 6, 6', 7\}$, such that for all ν-ary conditions d, all agents $X_\nu, x_{\nu+1} \in \Omega$, all actions $a \in A$, and all situations $\langle x, s\rangle$,

$$P_j d(X_\nu, x_{\nu+1}; x, s) \text{ iff [if } C^a_j d(X_\nu, x_{\nu+1}; x, s)\text{, then } a \text{ is prohibited for } x_{\nu+1}].$$

Note for example that $P_5 d(X_\nu, x_{\nu+1}; x, s)$ iff $P_{\mathrm{I,IV}} d(X_\nu, x_{\nu+1}; x, s)$. From the symmetry principles above it follows that

– $P_1 d(X_\nu, x_{\nu+1}; s)$ iff $P_1(d^\neg)(X_\nu, x_{\nu+1}; x, s)$
– $P_2 d(X_\nu, x_{\nu+1}; s)$ iff $P_4(d^\neg)(X_\nu, x_{\nu+1}; x, s)$
– $P_{2'} d(X_\nu, x_{\nu+1}; s)$ iff $P_{4'}(d^\neg)(X_\nu, x_{\nu+1}; x, s)$
– $P_5 d(X_\nu, x_{\nu+1}; s)$ iff $P_7(d^\neg)(X_\nu, x_{\nu+1}; x, s)$
– $P_6 d(X_\nu, x_{\nu+1}; s)$ iff $P_6(d^\neg)(X_\nu, x_{\nu+1}; x, s)$
– $P_{6'} d(X_\nu, x_{\nu+1}; s)$ iff $P_{6'}(d^\neg)(X_\nu, x_{\nu+1}; x, s)$

Now suppose that $P_j d(X_\nu, x_{\nu+1}; x, s)$ holds (is 'in effect') in situation $\langle x, s\rangle$, and that the corresponding transition type condition $C^a_j d(X_\nu, x_{\nu+1}; x, s)$ also holds for some action a and some agent $x_{\nu+1}$. Then a is prohibited for $x_{\nu+1}$: For all actions $a \in A$ and all agents $x_{\nu+1} \in \Omega$,

$$\begin{gathered} \mathit{Prohibited}_{x,s}(x_{\nu+1}, a) \text{ if there exists a condition } d, \\ \text{a sequence of agents } x_1, ..., x_\nu, \text{ and a } j \in \{2, 2', 4, 4', 5, 6, 6', 7\}, \text{ such that} \\ P_j d(x_1, ..., x_\nu, x_{\nu+1}; x, s) \ \&\ C^a_j d(x_1, ..., x_\nu, x_{\nu+1}; x, s). \end{gathered}$$

Table 2. Meaningful Combinations of Prohibited State Transition Types.

(I)	(II)	(III)	(IV)	$C_j^a d(X_\nu, x_{\nu+1}; x, s)$
-	-	-	-	-
-	-	X	-	$d(X_\nu; s) \wedge \neg d(X_\nu; a(x_{\nu+1}, s))$
-	-	-	X	$\neg d(X_\nu; s) \wedge \neg d(X_\nu; a(x_{\nu+1}, s))$
-	X	-	-	$\neg d(X_\nu; s) \wedge d(X_\nu; a(x_{\nu+1}, s))$
X	-	-	-	$d(X_\nu; s) \wedge d(X_\nu; a(x_{\nu+1}, s))$
-	-	X	X	$\neg d(X_\nu; a(x_{\nu+1}, s))$
-	X	X	-	$\neg(d(X_\nu; s) \leftrightarrow d(X_\nu; a(x_{\nu+1}, s)))$
X	-	-	X	$d(X_\nu; s) \leftrightarrow d(X_\nu; a(x_{\nu+1}, s))$
X	X	-	-	$d(X_\nu; a(x_{\nu+1}, s))$

2.2 Norm-Regulated Transition System Situations

A *norm-regulated transition system situation* is represented by an ordered pair $\langle \mathbf{S}, \mathcal{N} \rangle$ where $\mathbf{S} = \langle x, s, A, \Omega, S \rangle$ is a transition system situation and $\mathcal{N}$ is a normative system. We assume that (1) an event ε is of the form $x_{\nu+1}{:}a$ (i.e., represents an action a performed by an agent $x_{\nu+1}$; see Sect. 1.1) and (2) that norms apply to an individual agent $x_{\nu+1}$ in a state s. A norm in $\mathcal{N}$ is represented by an ordered pair $\langle G, C \rangle$, where the condition G on a situation $\langle x, s \rangle$ is the *ground* of the norm and the (normative) condition C on $\langle x, s \rangle$ is its *consequence*. (See, e.g., [22]) For example, $\langle g, P_j c \rangle$ represents the sentence

$$\forall x_1, x_2, ..., x_\nu, x_{\nu+1} \in \Omega : g(x_1, x_2, ..., x_p, x_{\nu+1}; x, s) \rightarrow \\ P_j c(x_1, x_2, ..., x_q, x_{\nu+1}; x, s)$$

where Ω is the set of agents, $x_{\nu+1}$ is the agent to which the norm applies, x is the 'moving' agent in the situation $\langle x, s \rangle$, and $\nu = \max(p, q)$. If the condition specified by the ground of a norm is true in some situation, then the (normative) consequence of the norm is in effect in that situation. To ensure that the agent $x_{\nu+1}$ to which the norm applies is the same as the moving agent x, we apply the 'move operator' M_i. This operator transforms a condition d on p agents in a state s to a condition $M_i d$ on $p+1$ agents in the situation $\langle x, s \rangle$, while at the same time identifying $x_{\nu+1}$ with x. (See [22, 7] for an explanation of the operator M_i.) If the normative system contains a norm whose ground holds in the the situation $\langle x, s \rangle$ and whose consequence prohibits the type of transition represented by the event $x_{\nu+1}{:}a$, then action a is prohibited for $x_{\nu+1}$ in $\langle x, s \rangle$:

$$Prohibited_{x,s}(x_{\nu+1}, a) \text{ according to } \mathcal{N}$$
$$\text{if there exists a condition } d \text{ and a condition } c \text{ and a } j \in \{2, 2', 4, 4', 5, 6, 6', 7\}$$
$$\text{such that } \langle M_i d, P_j c \rangle \text{ is a norm in } \mathcal{N}, \text{ and there exist } x_1, ..., x_\nu \text{ such that}$$
$$M_i d(x_1, ..., x_p, x_{\nu+1}; x, s) \;\&\; C_j^a c(x_1, ..., x_q, x_{\nu+1}; x, s), \text{ where } \nu = \max(p, q).$$

Since each situation for a Dalmas can be viewed as a transition system situation, it is straightforward to develop the Dalmas architecture (see Sect. 1.2) into an architecture for norm-regulated transition system situations. This means extending the set of seven type-operators T_i with corresponding E_i^a operators into a set of nine type-operators P_i with corresponding C_i^a operators, which calls for the definition of a structure similar to an *np-cis*[6]. The details are left for future work. The existing general-level Java/Prolog DALMAS implementation is easily adapted into a general-level implementation of norm-regulated transition system situations. In this framework, a norm is represented by a Prolog term `n/3` of the form `n(Id/N,OpG*G,OpC*C)`, where `Id` is an identifier of a norm-system and `N` is an identifier of an individual norm. `OpG*G` is a compound term representing an operator `OpG` applied to (the functor of) a state condition predicate `G`, forming the norm's ground. Similarly, `OpC*C` represents the norm's consequence.

2.3 Applications

The existing implementation of the **Colour & Form** DALMAS (see Sect. 1.2) has been adapted to serve as a demonstration of the use of norm-regulated transition system situations. The **Waste-collector** DALMAS and the **Forest Cleaner** DALMAS implementations may be adapted in a similar manner. However, the use of norm-regulated transition system situations is not limited to the DALMAS context. Many kinds of dynamic systems (including different types of transition systems and multi-agent systems) in which state transitions are connected to the actions of a single 'moving' agent, could be modelled and implemented by (iterated) use of a norm-regulated transition system situation. One example is the **Rooms** system, an implementation of (a variant of) the *Rooms* example by Craven and Sergot in [4, p. 178ff]. The example consists of a world in which agents of two categories ('male' and 'female') move around in a world of rooms that can contain any number of agents. Some rooms are connected by doorways (each connecting two rooms) through which the agents can pass[7], but only one agent at a time. The behaviour of the agents is regulated by a normative system stating that a female agent may not be alone in a room with a male agent. The restriction that only one agent at a time may move through a doorway is represented by the restriction that an event ε represents an action performed by a single agent x. To add some dynamics to the system, the behaviour of the agents is further governed by a simple utility function such that $left \succ_f stay \succ_f right$ and $right \succ_m stay \succ_m left$, where $\succ$ is the relation 'better than' and f and m stands for 'female' and 'male', respectively. Fig. 2 shows both a text-based and a graphical view of the initial state of the system, and the set of permissible acts for the acting agent f_1. The normative system contains the single norm $\langle M_0 opposite\_sex, P_7 alone\rangle$, which states that an agent may not act so that a pair of agents $\langle x_i, x_j\rangle$ such that x_i and x_j have opposite sex, end up alone in

[6] Normative-position condition-implication structure; see, e.g., [14, 22].

[7] More precisely, the agents may choose between three acts: $left$, $stay$ or $right$, but $left$ and $right$ are only feasible if there is a doorway in the corresponding direction. Note that the specific example in [4, p. 178ff] has one female and two male agents and two rooms, while the **Rooms** system has three rooms.

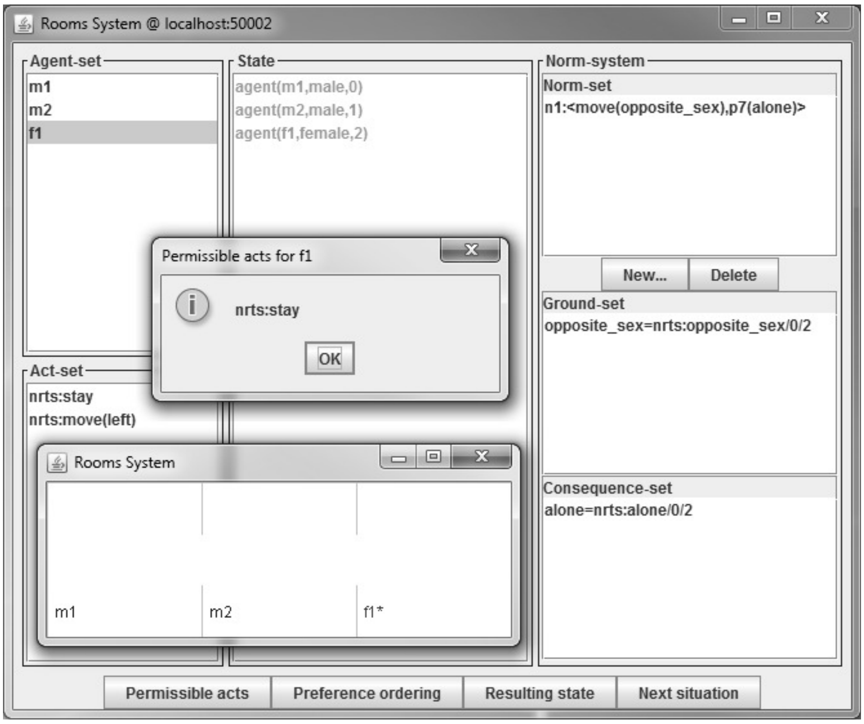

Fig. 2. Screenshot: Initial situation of a **Rooms** system execution

the same room. This includes moving to a room containing a single agent of the opposite sex as well as leaving two other agents of opposite sex alone in the same room. We see that of the two feasible acts *stay* and $left$ in the current situation, only *stay* is permissible according to the normative system, since if f_1 moves left she ends up alone with m_2.

The source code for the **Colour and Form** system and the **Rooms** system, as well as for the general-level Java/Prolog implementation of norm-regulated transition system situations is available for download[8] and is publicly and freely disseminated. The example systems are quite simple, but nicely illustrate some features of iterated use of norm-regulated transition system situations, e.g. the ability to investigate the interplay between a normative system that determines the scope of permissible actions for agents and utility functions that represent the preferences of the agents. They demonstrate that the general-level Java/Prolog implementation can be used as a tool for the implementation of such systems. The framework includes a Prolog logic server as a backend and (if desired) a Java user interface as frontend, functioning as a lookup-service that answers questions such as 'is act a permissible for x in state s, according to normative system $\mathcal{N}$' or 'which acts are permissible for x in state s, according to $\mathcal{N}$'. At the

[8] http://drp.name/norms/nrtssit

system level, it could be used to maintain a normative system for some society, in combination with some norm enforcement strategy. At the agent level, it could be used as a common normative framework that is shared by individual agents that take norms into account in their reasoning cycle, or as part of an agent's internal architecture, either to represent a model of society's normative system or to represent an agent's 'internal' normative system ('ethics'). Naturally, the use of both Java and Prolog as implementation languages has both advantages and disadvantages. The primary advantage is that this approach combines the strengths of two different programming paradigms and languages. On the other hand, it demands skills in both object-oriented and logic programming of the developer wishing to use the framework to develop a specific system.

Remark 1. Regarding computational complexity, it can be noted that the framework works well for the simple systems discussed here, but certainly has room for various performance optimizations. Still, even with such optimizations made, scaleability will remain a challenge for this framework as well as for most other frameworks for norm-regulated multi-agent systems (see, e.g., [24, p. 52]), since the time to test each norm is in the worst case roughly proportional to n^{ν}, where n is the size of the agent set Ω, $\nu = \max(p, q)$ and p and q is the arity of the ground (resp. consequence) of the norm.[9] One way to, at least partially, address these issues is to explore the possibility to express a normative system in an economic way by its set of 'minimal norms' (see for example [22, 17]).

3 Conclusion and Future Work

This paper has introduced the notion of a transition system situation, which is intended to represent a single step in the run of many kinds of transition systems. In a norm-regulated transition system situation, the permission or prohibition of actions is related to the permission or prohibition of different types of state transitions with respect to some condition d on a number of agents $x_1, \ldots, x_\nu$ in a state. The framework uses a representation of conditional norms based on the algebraic approach[10] to normative systems used in [22] and a systematic exploration of the possible types of state transitions with respect to $d(x_1, \ldots, x_\nu)$.

By adaption of the existing implementation of the Dalmas architecture, a general-level Java/Prolog framework for norm-regulated transition system situations (together with some simple example systems) has been developed. The set of eight transition type conditions C_i^a is an extension of the set of six E_i^a conditions in [22]. These conditions were intended as an interpretation in the Dalmas context of Lindahl's set of one-agent types of normative positions. The (potential) connection between the combination of P_i and C_i^a and the Kanger-Lindahl

[9] As noted in [21, p. 31], the computational complexity of any specific implementation may be more formally analysed through algorithmic analysis, e.g. average-case analysis.

[10] This approach was originally developed in a series of papers; see for example [13, 15–17].

theory of normative positions is interesting. It has been partly investigated in [8], but deserves to be further explored.

Lindahl and Odelstad argue that a normative system should express "... general rules where no individual names occur. If the task is to represent a normative system this feature of generality has to be taken into account." [17, p. 5] An advantage of their algebraic approach to normative systems, besides for example efficient automation and mechanization (see, e.g., [25, p. 197] with references), is in fact the expressive power it yields. The algebraic normative framework presented in this paper allows the construction of norms based on conditions on an arbitrary number of agents, in contrast to for example Dynamic deontic logic [20] and Dynamic logic of permission [19] which both have their roots in Propositional Dynamic Logic (PDL). Unlike in the agent-stranded coloured transition systems [4, 24], the framework presented in this paper does not explicitly distinguish between state permission laws and action permission laws. It allows, however, a state permission law to be represented implicitly as a special case, by a norm which prohibits all transitions that lead to an undesired state. Our framework treats all norms as action permission laws, in the sense that actions are prohibited in different states as a consequence of certain transition types being prohibited by the normative system. It allows the creation of norms that forbid specific named actions in certain situations, by choosing a normative consequence that forbids the agent to act so that it ends up in a state where the last action performed was the prohibited action. This requires some sort of history of actions to be part of the state of the system.

The idea to base norms on permissible and prohibited types of state transitions has, to the author's knowledge, not been systematically explored before. It appears that the language for action permission laws used by Craven and Sergot also allows the formulation of norms that prohibit certain types of transitions, but an example of this feature is not given in [4]. In Dynamic deontic logic it is only the state resulting from a transition that determines if the transition is classified as 'permitted/non-permitted', while in Dynamic logic of permission, it is executions of actions that are classified as 'permitted/non-permitted'. van der Meyden's treatment of permission uses the process semantics for actions, in which the denotation of an action expressions is a set of sequences of states. This allows for the description of the states of affairs during the execution of an action; the permission of an action is not dependent only on the state resulting from the execution of the action, but also on the intermediate states.

The systematic treatment of the different types of transitions ensures that the set of transition type operators C_j^a and the corresponding prohibition operators P_j exhaust the space of meaningful transition type prohibitions. Therefore, norm-regulated transition system situations could be used in a given problem domain to systematically search for the 'best' normative system for (a class of) dynamic systems, according to some criteria for evaluation of the system's performance. For example, as suggested in [8], a genetic algorithm or some other mechanism from machine learning could be employed to seek the optimal normative system for a particular task. This requires some mechanism for *norm* *change*. In the current architecture, norms may be changed 'from the outside', but not 'from the inside' as a consequence of an action by an agent in a state s, since the normative system $\mathcal{N}$ is not itself considered a part of s. An interesting line of future work is to explore the possibility to let the normative system be a part of the state, thereby letting agents choose actions that modify the normative system. Norm change is another area in which the notion of 'minimal norms' may be of special significance, as suggested in [17, Sect. 2.1.2 and 4.3].

The requirement that each event ε in a norm-regulated transition system situation represents an action performed by a single agent deserves further attention. It corresponds roughly to the restriction in the *Rooms* example (Sect. 2.3) that only one agent at a time can move through a doorway. This raises a number of questions regarding the relationship between norm-regulated transition system situations and transition systems in which a single transition may correspond to the simultaneous action of several agents, possibly including 'actions' by the environment itself. These issues deserve a deeper discussion, which is left for future papers.

Another interesting issue is *consistency*. An inconsistent normative system may lead to a situation in which the deontic structure is empty, i.e. all actions are prohibited. How the system should behave in such a situation is heavily dependent on the nature of the specific application at hand; this is not specified by the general-level framework.

Acknowledgements

The author wishes to thank Jan Odelstad and Magnus Boman for valuable ideas and suggestions, participants of ICAART 2013 for discussions in relation to this paper and the organizers of the conference for the opportunity to expand my conference paper.

References

1. Cliffe, O., De Vos, M., Padget, J.: Specifying and analysing agent-based social institutions using answer set programming. In: Boissier, O., Padget, J., Dignum, V., Lindemann, G., Matson, E., Ossowski, S., Sichman, J., Vázquez-Salceda, J. (eds.) Coordination, Organizations, Institutions, and Norms in Multi-Agent Systems, Lecture Notes in Computer Science, vol. 3913, pp. 99–113. Springer Berlin / Heidelberg (2006), doi:10.1007/11775331_7
2. Cliffe, O., De Vos, M., Padget, J.: Answer set programming for representing and reasoning about virtual institutions. In: Inoue, K., Satoh, K., Toni, F. (eds.) Computational Logic in Multi-Agent Systems, Lecture Notes in Computer Science, vol. 4371, pp. 60–79. Springer Berlin / Heidelberg (2007), doi:10.1007/978-3-540-69619-3_4
3. Combettes, S., Hanachi, C., Sibertin-Blanc, C.: Organizational petri nets for protocol design and enactment. In: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems. pp. 1384–1386. AAMAS '06, ACM, New York, NY, USA (2006), doi:10.1145/1160633.1160892

4. Craven, R., Sergot, M.: Agent strands in the action language nC+. Journal of Applied Logic 6(2), 172–191 (2008), selected papers from the 8th International Workshop on Deontic Logic in Computer Science, 8th International Workshop on Deontic Logic in Computer Science. doi:10.1016/j.jal.2007.06.007
5. Gaertner, D., Clark, K., Sergot, M.: *Ballroom etiquette:* a case study for norm-governed multi-agent systems. In: Noriega, P., Vázquez-Salceda, J., Boella, G., Boissier, O., Dignum, V., Fornara, N., Matson, E. (eds.) Coordination, Organizations, Institutions, and Norms in Agent Systems II, Lecture Notes in Computer Science, vol. 4386, pp. 212–226. Springer Berlin / Heidelberg (2007), doi:10.1007/978-3-540-74459-7_14
6. Governatori, G., Rotolo, A., Sartor, G.: Temporalised normative positions in defeasible logic. In: Proceedings of the 10th international conference on Artificial intelligence and law. pp. 25–34. ICAIL '05, ACM, New York, NY, USA (2005), doi:10.1145/1165485.1165490
7. Hjelmblom, M.: Deontic action-logic multi-agent systems in Prolog. Tech. Rep. 30, University of Gävle, Division of Computer Science (2008), urn:nbn:se:hig:diva-1475
8. Hjelmblom, M.: State transitions and normative positions within normative systems. Tech. Rep. 37, University of Gävle, Department of Industrial Development, IT and Land Management (2011), urn:nbn:se:hig:diva-10595
9. Hjelmblom, M.: Norm-regulated transition system situations. In: Filipe, J., Fred, A. (eds.) Proceedings of the 5th International Conference on Agents and Artificial Intelligence. pp. 109–117. ICAART 2013, SciTePress, Portugal (2013)
10. Hjelmblom, M., Odelstad, J.: jDALMAS: A Java/Prolog framework for deontic action-logic multi-agent systems. In: Håkansson, A., Nguyen, N., Hartung, R., Howlett, R., Jain, L. (eds.) Agent and Multi-Agent Systems: Technologies and Applications, Lecture Notes in Computer Science, vol. 5559, pp. 110–119. Springer Berlin / Heidelberg (2009), doi:10.1007/978-3-642-01665-3_12
11. Laaksolahti, J., Boman, M.: Anticipatory guidance of plot. CoRR cs.AI/0206041 (2002), doi:10.1007/978-3-540-45002-3_14
12. Lindahl, L.: Position and change: a study in law and logic. Synthese library, D. Reidel Pub. Co. (1977), http://www.google.com/books?id=_QwWhOK8aY0C
13. Lindahl, L., Odelstad, J.: Normative systems and their revision: An algebraic approach. Artificial Intelligence and Law 11, 81–104 (2003), doi:10.1023/B:ARTI.0000046005.10529.47
14. Lindahl, L., Odelstad, J.: Normative positions within an algebraic approach to normative systems. Journal of Applied Logic 2(1), 63 – 91 (2004), the Sixth International Workshop on Deontic Logic in Computer Science. doi:10.1016/j.jal.2004.01.004
15. Lindahl, L., Odelstad, J.: Intermediaries and intervenients in normative systems. Journal of Applied Logic 6(2), 229 – 250 (2008), selected papers from the 8th International Workshop on Deontic Logic in Computer Science, 8th International Workshop on Deontic Logic in Computer Science. doi:10.1016/j.jal.2007.06.010
16. Lindahl, L., Odelstad, J.: Stratification of normative systems with intermediaries. Journal of Applied Logic 9(2), 113 – 136 (2011), special Issue: Selected and revised papers from the Ninth International Conference on Deontic Logic in Computer Science (DEON 2008), Ninth International Conference on Deontic Logic in Computer Science. doi:10.1016/j.jal.2010.01.002
17. Lindahl, L., Odelstad, J.: The Theory of Joining-Systems, vol. 1, pp. 547–636. College Publications, London (2012), to appear in Gabbay; Horthy; van der Meyden; and van der Torre: Handbook of Normative systems

18. Makinson, D., van der Torre, L.: What is input/output logic? input/output logic, constraints, permissions. In: Boella, G., van der Torre, L., Verhagen, H. (eds.) Normative Multi-agent Systems. No. 07122 in Dagstuhl Seminar Proceedings, Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany, Dagstuhl, Germany (2007), http://drops.dagstuhl.de/opus/volltexte/2007/928
19. van der Meyden, R.: The dynamic logic of permission. Journal of Logic and Computation 6(3), 465–479 (June 01 1996)
20. Meyer, J.J.C.: A different approach to deontic logic: deontic logic viewed as a variant of dynamic logic. Notre Dame Journal of Formal Logic 29(1), 109–136 (1987), doi:10.1305/ndjfl/1093637776
21. Odelstad, J.: Many-Sorted Implicative Conceptual Systems. Ph.D. thesis, Royal Institute of Technology, Computer and Systems Sciences, DSV (2008), qC 20100901
22. Odelstad, J., Boman, M.: Algebras for agent norm-regulation. Annals of Mathematics and Artificial Intelligence 42, 141–166 (2004), doi:10.1023/B:AMAI.0000034525.49481.4a
23. Raskin, J.F., van der Torre, L.W., Tan, Y.H.: How to model normative behavior in petri nets. In: Proceedings of the 2nd Modelage Workshop on Formal Models of Agents. pp. 223–241 (1996), http://hdl.handle.net/2013/ULB-DIPOT:oai:dipot.ulb.ac.be:2013/70564
24. Sergot, M.: Action and agency in norm-governed multi-agent systems. In: Artikis, A., O'Hare, G., Stathis, K., Vouros, G. (eds.) Engineering Societies in the Agents World VIII, Lecture Notes in Computer Science, vol. 4995, pp. 1–54. Springer Berlin / Heidelberg (2008), doi:10.1007/978-3-540-87654-0_1
25. Solin, K.: Modal semirings with operators for knowledge representation. In: Filipe, J., Fred, A. (eds.) Proceedings of the 5th International Conference on Agents and Artificial Intelligence. pp. 197–202. ICAART 2013, SciTePress, Portugal (2013)
26. Vázquez-Salceda, J., Aldewereld, H., Dignum, F.: Implementing norms in multiagent systems. In: Lindemann, G., Denzinger, J., Timm, I., Unland, R. (eds.) Multiagent System Technologies, Lecture Notes in Computer Science, vol. 3187, pp. 313–327. Springer Berlin / Heidelberg (2004), doi:10.1007/978-3-540-30082-3_23