Normative Systems in Computer Science
Ten Guidelines for Normative Multiagent Systems

Guido Boella[1,2], Gabriella Pigozzi[2], and Leendert van der Torre[2]

[1] Department of Computer Science, University of Torino
[2] Computer Science and Communication, University of Luxembourg

Abstract. In this paper we introduce and discuss ten guidelines for the use of normative systems in computer science. We adopt a multiagent systems perspective, because norms are used to coordinate, organize, guide, regulate or control interaction among distributed autonomous systems. The first six guidelines are derived from the computer science literature. From the so-called 'normchange' definition of the first workshop on normative multiagent systems in 2005 we derive the guidelines to motivate which definition of normative multiagent system is used, to make explicit why norms are a kind of (soft) constraints deserving special analysis, and to explain why and how norms can be changed at runtime. From the so-called 'mechanism design' definition of the second workshop on normative multiagent systems in 2007 we derive the guidelines to discuss the use and role of norms as a mechanism in a game-theoretic setting, clarify the role of norms in the multiagent system, and to relate the notion of "norm" to the legal, social, or moral literature. The remaining four guidelines follow from the philosophical literature: use norms also to resolve dilemmas, and in general to coordinate, organize, guide, regulate or control interaction among agents, distinguish norms from obligations, prohibitions and permissions, use the deontic paradoxes only to illustrate the normative multiagent system, and consider regulative norms in relation to other kinds of norms and other social-cognitive computer science concepts.

1 Introduction

Normative systems are "systems in the behavior of which norms play a role and which need normative concepts in order to be described or specified" [36, preface]. There is an increasing interest in normative systems in the computer science community, due to the observation five years ago in the so-called AgentLink Roadmap [33, Fig. 7.1], a consensus document on the future of multiagent systems research, that norms must be introduced in agent technology in the medium term (i.e., now!) for infrastructure for open communities, reasoning in open environments and trust and reputation. However, there is no consensus yet in the emerging research area of normative multiagent systems on the kind of norms to be used, or the way to use them. Consider the following lines taken from a paper review report. A norm like "You should empty your plate" may be criticized, because it is not a (generic) norm but an obligation, or a sentence not presented as a norm, such as an imperative or command like "Empty your plate!", may be criticized because it is a norm. Alternatively, a proposed normative multiagent systems may be criticized by a reviewer, because, for example, norms cannot be violated, norms cannot be changed, and so on. These criticisms suggest that more agreement on the use of norms and normative systems in computer science would be useful.

The research question of this paper is to give general guidelines for the use of "norms" and "normative systems" in computer science. During the past two decades normative systems have been studied in a research field called deontic logic in computer science (ΔEON), and normative multiagent systems may be seen as the research field where the traditional normative systems and ΔEON meet agent research. In these areas, the following two related challenges emerged to a common use of "norms" and "normative systems" in computer science.

There are many distinct notions of "normative systems" in the literature due to the use of the concept "norm" in distinct disciplines, just like there are many definitions of "agent" or "actor" due to its use across disciplines. Traditionally normative systems have been studied in philosophy, sociology, law, and ethics, and "norms" can therefore be, for example, social expectations, legal laws or linguistic imperatives or commands.

The role of norms in computer science is changing and solutions based on multiagent systems are increasing. The seventh ΔEON conference [31, 32] in 2004 in Madeira, Portugal, had as special theme "deontic logic and multiagent systems," the eighth ΔEON conference in 2006 in Utrecht, the Netherlands, had as special focus "artificial normative systems" [22, 21], and the ninth ΔEON conference [22, 43] in Luxembourg in 2008 was co-located with the third workshop on normative multiagent systems NorMAS. Gradually the ΔEON research focus changes from logical relations among norms to, for example, agent decision making, and to systems in which norms are created and in which agents can play the role of legislators.

We approach this question of defining guidelines for normative multiagent system research by first considering two consensus definitions in the computer science literature of previous normative multiagent systems NorMAS workshops, from which we derive our first six guidelines. The remaining four guidelines follow from a short survey of the philosophical literature.

2 Normative multiagent systems

Before we consider the 'normchange' and 'mechanism design' definition of normative multiagent systems, we start with a dictionary definition of normative systems. With 'normative' we mean 'conforming to or based on norms', as in *normative behavior* or *normative judgments*. According to the Merriam-Webster Online [35] Dictionary, other meanings of normative not considered here are 'of, relating to, or determining norms or standards', as in *normative tests*, or 'prescribing norms', as in *normative rules of ethics* or *normative grammar*. With 'norm' we mean 'a principle of right action binding upon the members of a group and serving to guide, control, or regulate proper and acceptable behavior'. Other meanings of 'norm' given by the Merriam-Webster Online Dictionary but not considered here are 'an authoritative standard or model', 'an average like a standard, typical pattern, widespread practice or rule in a group', and various definitions used in mathematics.

2.1 The normchange definition

The first definition of a normative multiagent system emerged after two days of discussion at the first workshop on normative multiagent systems NorMAS held in 2005 as a symposium of the Artificial Intelligence and Simulation of Behaviour convention (AISB) in Hatfield, United Kingdom:

The normchange definition. "A normative multiagent system is a multiagent system together with normative systems in which agents on the one hand can decide whether to follow the explicitly represented norms, and on the other the normative systems specify how and in which extent the agents can modify the norms" [8].

The first three guidelines are derived from this definition. The first one concerns the explicit representation of norms, which has been interpreted either that norms must be explicitly represented in the system (the 'strong' interpretation) or that norms must be explicitly represented in the system specification (the 'weak' interpretation). The first guideline is to make explicit and motivate which interpretation is used, the strong one, the weak one, or none of them.

Guideline 1 *Motivate which definition of normative multiagent system is used.*

The motivation for the strong interpretation of the explicit representation is to prevent a too general notion of norms. Any requirement can be seen as a norm the system has to comply with; but why should we do so? Calling every requirement a norm makes the concept empty and useless. The weak interpretation is used to study the following two important problems in normative multiagent systems.

Norm compliance. How to decide whether systems or organizations comply with relevant laws and regulations? For example, is a hospital organized according to medical regulations? Does a bank comply with Basel 2 regulations?

Norm implementation. How can we design a system such that it complies with a given set of norms? For example, how to design an auction such that agents cannot cooperate?

The second guideline follows from the fact that agents can decide whether to follow the norms. This part of the definition is borrowed from the ΔEON tradition, whose founding fathers Meyer and Wieringa observe that "until recently in specifications of systems in computational environments the distinction between normative behavior (as it *should be*) and actual behavior (as it *is*) has been disregarded: mostly it is not possible to specify that some system behavior is non-normative (illegal) but nevertheless possible. Often illegal behavior is just ruled out by specification, although it is very important to be able to specify what should happen if such illegal but possible behaviors occurs!" [36, preface]. However, constraints are well studied and well understood concepts, so if a norm is a kind of constraint, the question immediately is raised what is special about them.

Guideline 2 *Make explicit why your norms are a kind of (soft) constraints that deserve special analysis.*

Examples of issues which have been analyzed for norms but to a less degree for other kinds of constraints are ways to deal with violations, representation of permissive norms, the evolution of norms over time (in deontic logic), the relation between the cognitive abilities of agents and the global properties of norms, how agents can acquire norms, how agents can violate norms, how an agent can be autonomous [17] (in normative agent architectures and decision making), how norms are created by a legislator, emerge spontaneously or are negotiated among the agents, how norms are enforced, how constitutive or counts-as norms are used to describe institutions, how norms are related to other social and legal concepts, how norms structure organizations, how norms coordinate groups and societies, how contracts are related to contract frames and contract law, how legal courts are related, and how normative systems interact?

For example, the norms of global policies may be represented as soft constraints, which are used in detective control systems where violations can be detected, instead of hard constraints restricted to preventative control systems in which violations are impossible. The typical example of the former is that you can enter a train without a ticket, but you may be checked and sanctioned, and an example of the latter is that you cannot enter a metro station without a ticket. However, if the norms are represented as constraints, then how to analyze that detective control is the result of actions of agents and therefore subject to errors and influenceable by actions of other agents? For example, it may be the case that violations are not often enough detected, that law enforcement is lazy or can be bribed, there are conflicting obligations in the normative system, that agents are able to block the sanction, block the prosecution, update the normative system, etc.

The third guideline follows from the fact that norms can be changed by the agents or the system, which distinguished this definition of normative multiagent system from the common framework used in the ΔEON community, and led to the identification of this definition as the "normchange" definition of normative multiagent systems.

Guideline 3 *Explain why and how norms can be changed at runtime.*

For example, a norm can be made by an agent, as legislators do in a legal system, or there can be an algorithm that observes agent behavior, and suggests a norm when it observes a pattern. The agents can vote on the acceptance of the norm. Likewise, if the system observes that a norm is often violated, then apparently the norm does not work as desired, and it undermines the trust of the agents in the normative system, so the system can suggest that the agents can vote whether to retract or change the norm.

2.2 The mechanism design definition

The fourth, fifth and sixth guideline follow from the consensus definition of the second workshop on normative multiagent systems NorMAS held as Dagstuhl Seminar 07122 in 2007. After four days of discussion, the participants agreed to the following consensus definition:

The mechanism design definition. "A normative multiagent system is a multiagent system organized by means of mechanisms to represent, communicate, distribute, detect, create, modify, and enforce norms, and mechanisms to deliberate about norms and detect norm violation and fulfilment." [10]

The fourth guideline emphasizes the game-theoretic model and the notion of a norm as a mechanism. According to Boella *et al.*, "the emphasis has shifted from representation issues to the mechanisms used by agents to coordinate themselves, and in general to organize the multiagent system. Norms are communicated, for example, since agents in open systems can join a multiagent system whose norms are not known. Norms are distributed among agents, for example, since when new norms emerge the agent could find a new coalition to achieve its goals. Norm violations and norm compliance are detected, for example, since spontaneous emergence norms of among agents implies that norm enforcement cannot be delegated to the multiagent infrastructure." [10]

Guideline 4 *Discuss the use and role of norms always as a mechanism in a game-theoretic setting.*

Here we refer to game theory in a very liberal sense, not only to classical game theory studied in economics, which has been criticized for its ideality assumptions. Of particular interest are alternatives taking the limited or bounded rationality of decision makers into account. For example, Newell [37] and others develop theories in artificial intelligence and agent theory, replace probabilities and utilities by informational (knowledge, belief) and motivational attitudes (goal, desire), and the decision rule by a process of deliberation. Bratman [11] further extends such theories with intentions for sequential decisions and norms for multiagent decision making. Alternatively, Gmytrasiewitcz and Durfee [19] replace the equilibria analysis in game theory by recursive modelling, which considers the practical limitations of agents in realistic settings such as acquiring knowledge and reasoning so that an agent can build only a finite nesting of models about other agents' decisions.

Games can explain that norms should satisfy various properties to be effective as a mechanism to obtain desirable behavior. For example, the system should not sanction without reason, as for example Caligula or Nero did in the ancient Roman times, as the norms would loose their force to motivate agents. Moreover, sanctions should not be too low, but they also should not be too high, as shown by argument of Beccaria. Otherwise, once a norm is violated, there is no way to prevent further norm violations.

Games can explain also the role of various kinds of norms in a system. For example, assume that norms are added to the system one after the other and this operation is performed by different authorities at different levels of the hierarchy. Lewis "master and slave" game [30] shows that the notion of permission alone is not enough to build a normative system, because only obligations divide the possible actions into two categories or spheres: the sphere of prohibited actions and the sphere of permitted (i.e., not forbidden) actions or "the sphere of permissibility". More importantly, Bulygin [13] explains why permissive norms are needed in normative systems using his "Rex, Minister and Subject" game. "Suppose that Rex, tired of governing alone, decides one day to appoint a Minister and to endow him with legislative power. [...] an action commanded by Minister becomes as obligatory as if it would have been commanded by Rex. But Minister has no competence to alter the commands and permissions given by Rex." If Rex permits hunting on Saturday and then Minister prohibits it for the whole week, its prohibition on Saturday remains with no effect.

As another example, in our game theoretic approach to normative systems [9] we study the following kind of normative games.

Violation games: interacting with normative systems, obligation mechanism, with applications in trust, fraud and deception.

Institutionalized games: counts-as mechanism, with applications in distributed systems, grid, p2p, virtual communities.

Negotiation games: MAS interaction in a normative system, norm creation action mechanism, with applications in electronic commerce and contracting.

Norm creation games: multiagent system structure of a normative system, permission mechanism, with applications in legal theory.

Control games: interaction among normative systems, nested norms mechanism, with applications in security and secure knowledge management systems.

The fifth guideline follows from the introduction of organizational issues in the definition of normative multiagent systems. Norms are no longer seen as the mechanism to regulate behavior of the system, but part of a larger institution. This raises the question what precisely the role of norms is in such an organization.

Guideline 5 *Clarify the role of norms in your system.*

Norms are rules used to guide, control, or regulate desired system behavior. However, this is not unproblematic. For example, consider solving traffic problems by introducing norms, as a cheap alternative to building new roads. It does not work, for the following two reasons. The first reason is that if you change the system by building new norms or introducing new norms, then people will adjust their behavior. For example, when roads improve, people tend to live further away from their work. In other words, a normative multiagent system is a self-organizing system. Moreover, the second problem with norm design is that norms can be violated. For example, most traffic is short distance, for which we could forbid using the car. However, it is hard to enforce such a norm, since people will always claim to have come from long distance, even if they live around the corner.

Norms can also be seen as one of the possible incentives to motivate agents, which brings us again back to economics.

> "Economics is, at root, the study of incentives: how people get what they want, or need, especially when other people want or need the same thing. Economists love incentives. They love to dream them up and enact them, study them and tinker with them. The typical economist believes the world has not yet invented a problem that he cannot fix if given a free hand to design the proper incentive scheme. His solution may not always be pretty–but the original problem, rest assured, will be fixed. An incentive is a bullet, a lever, a key: an often tiny object with astonishing power to change a situation.
> ...
> There are three basic flavors of incentive: economic, social, and moral. Very often a single incentive scheme will include all three varieties. Think about the anti-smoking campaign of recent years. The addition of $3-per-pack "sin tax" is a strong economic incentive against buying cigarettes. The banning of cigarettes in restaurants and bars is a powerful social incentive. And when the U.S. government asserts that terrorists raise money by selling black-market cigarettes, that acts as a rather jarring moral incentive.' [29]

Here it is important to see that moral incentives are very different from financial incentives. For example, Levitt [29, p.18-20], discussing an example of Gneezy and Rustichini [20], explains that the number of violations may *increase* when financial sanctions are imposed, because the moral incentive to comply with the norm is destroyed. The fact that norms can be used as a mechanism to obtain desirable system behavior, i.e. that norms can be used as incentives for agents, implies that in some circumstances economic incentives are not sufficient to obtain such behavior. For example, in a widely discussed example of the so-called centipede game, there is a pile of thousand pennies, and two agents can in turn either take one or two pennies. If an agent takes one then the other agent takes turn, if it takes two then the game ends. A backward induction argument implies that it is rational only to take two at the first turn. Norms and trust have been discussed to analyze this behavior, see [28] for a discussion.

A rather different role of norms is to organize systems. To manage properly complex systems like multiagent systems, it is necessary that they have a modular design. While in traditional software systems, modularity is addressed via the notions of class and object, in multiagent systems the notion of organization is borrowed from the ontology of social systems. Organizing a multiagent system allows to decompose it and defining different levels of abstraction when designing it. Norms are another answer to the question of how to model organizations as first class citizens in multiagent systems. Norms are not usually addressed to individual agents, but rather they are addressed to roles played by agents [6]. In this way, norms from a mechanism to obtain the behavior of agents, also become a mechanism to create the organizational structure of multiagent systems. The aim of an organizational structure is to coordinate the behavior of agents so to perform complex tasks which cannot be done by individual agents. In organizing a system all types of norms are necessary, in particular, constitutive norms, which are used to assign powers to agents playing roles inside the organization. Such powers allow to give commands to other agents, make formal communications and to restructure the organization itself, for example, by managing the assignment of agents to roles. Moreover, normative systems allow to model also the structure of an organization and not only the interdependencies among the agents of an organization. Consider a simple example from organizational theory in Economics: an enterprise which is composed by a direction area and a production area. The direction area is composed by the CEO and the board. The board is composed by a set of administrators. The production area is composed by two production units; each production unit by a set of workers. The direction area, the board, the production area and the production units are functional areas. In particular, the direction area and the production areas belong to the organization, the board to the direction area, etc. The CEO, the administrators and the members of the production units are roles, each one belonging to a functional area, e.g., the CEO is part of the direction area. This recursive decomposition terminates with roles: roles, unlike organizations and functional areas, are not composed by further social entities. Rather, roles are played by other agents, real agents (human or software) who have to act as expected by their role. Each of these elements can be seen as an institution in a normative system, where legal institutions are defined by Ruiter [39] as "systems of [regulative and constitutive] rules that provide frameworks for social action within larger rule-governed settings". They are "relatively independent institutional legal orders within the comprehensive legal orders".

The sixth guideline follows from the trend towards a more dynamic interactionist view identified at the second NorMAS workshop. "This shift of interest marks the passage of focus from the more static legalistic view of norms (where power structures are fixed) to the more dynamic interactionist view of norms (where agent interaction is the base for norm related regulation)." This ties in to what Strauss [42] called "negotiated order", Goffman's [23] view on institutions, and Giddens' [18] structuration theory. The two views are summarized in Table 1. For example, if in a normative system the norms are created by agents it is more a legalistic view, but if there is an algorithm that observes behavior and proposes norms, it is more an interactionist view. The latter procedure can still be put up to vote for the agents, and being accepted or rejected. As another example, suppose a monitoring system observes that some norms are violated frequently, then it can propose to delete the norms, for example because the violations decrease the trust of the agents in the system.

	Legalistic view	Interactionist view
	top-down view	bottom-up view
		autonomous individually oriented view
normative system	regulatory instrument	regularities of behavior
	to regulate emerging behavior of open systems	emerge without any enforcement system
compliance	sanctions	sharing of the norms
		their goals happen to coincide
		they feel themselves as part of the group
		they share the same values
		sanctions are not always necessary
		social blame and spontaneous exclusion
freedom to create norms	restricted to contracts	emergence of norms

Table 1. Two views on normative multiagent systems

Guideline 6 *Relate the notion of "norm" to the legal, social, or moral literature.*

Boella *et al.* put the legalistic and interactionist view in the context of five levels in the development of normative multiagent systems, summarized in Table 2. They observe that "for each level the development of the normative multiagent system will take a much larger effort than the development of similar systems at lower levels." For example, if norms are explicitly represented (level 2) rather than built into the system (level 1), then the system has to be much more flexible to deal with the variety of normative systems that may emerge. However, it may be expected that normative multiagent systems realized at higher levels will have a huge effect on social interaction, in particular on the web" [10]. We illustrate the more dynamic interactionist viewpoint on normative multiagent systems using virtual communities in virtual reality settings like Second Life. In these virtual communities, human agents interact with artificial agents in a virtual world. This interactionist view, which has been promoted in the multiagent systems community by Cristiano Castelfranchi [14], becomes essential in applications related to virtual communities. In Second Life, for example, communities emerge in which the behavior of its members show increasing homogeneity.

level		
1	off-line norm design [41]	norms are imposed by the designer and automatically enforced, agents cannot organize themselves by means of norms
2	norm representation	norms are explicitly represented they can be used in agent communication and negotiation a simple kind of organizations and institutions can be created
3	norm manipulation	a legal reality is created agents can add and remove norms following the rules of the normative system
4	social reality	the ten challenges discussed Table 3
5	moral reality	This goes beyond present studies in machine ethics [4]

Table 2. Five levels in the development of normative multiagent systems. [10]

Boella *et al.* also mention ten challenges posed by the interactionist viewpoint: They “take the perspective from an agent programmer, and consider which kinds of tools like programming primitives, infrastructures, protocols, and mechanisms she needs to deal with norms in the example scenario. Similar needs exist at the requirements analysis level, or the design level, but we have chosen for the programming level since it makes the discussion more concrete, and this level is often ignored when norms are discussed. The list is not exhaustive, and there is some overlap between the challenges. Our aim is to illustrate the range of topics which have to be studied, and we therefore do not attempt to be complete” [10].

Challenge	Tool
1	Tools for agents supporting communities in their task of recognizing, creating, and communicating norms to agents
2	Tools for agents to simplify normative systems, recognize when norms have become redundant, and to remove norms
3	Tools for agents to enforce norms
4	Tools for agents to preserve their autonomy
5	Tools for agents to construct organizations
6	Tools for agents to create intermediate concepts and normative ontology, for example to decide about normative gaps
7	Tools for agents to decide about norm conflicts
8	Tools for agents to voluntarily give up some norm autonomy by allowing automated norm processing in agent acting and decision making
9	Tools for conviviality
10	Tools for legal responsibility of the agents and their principals

Table 3. Ten challenges posed by the interactionist viewpoint. [10]

3 Philosophical foundations

We consider only four guidelines from the rich history of deontic logic in philosophical logic. The first two guidelines follow from the history of deontic logic, the third guideline from the methodology in deontic logic based on deontic paradoxes, and the fourth guideline from the deontic logic in computer science to study norms in the way they interact with other concepts. We believe philosophical logic has much more to offer for computer scientists, but we restrict ourselves to the most important issues.

3.1 Deontic logic

In 1951, the philosopher and logician Von Wright wrote a paper called "deontic logic" [45], which subsequently became the name of the research area concerned with normative concepts such as obligation, prohibition and permission. The term deontic is derived from the ancient Greek déon, meaning that which is binding or proper. The basis of his formal system was an observed relation between obligation and permission. For example, he defined the obligation to tell the truth by interpreting that it is good to tell the truth, and therefore it is bad to lie. If it is bad to lie then it is forbidden to lie, and therefore it is not permitted to lie. Summarizing, something is obligatory when its absence is not permitted. This logical relation is based on the binary distinction between good and bad, as illustrated by its possible worlds semantics distinguishing between good and bad worlds.

The relation between obligation and violation was given by Anderson seven years later in 1958, in a paper called "A Reduction of Deontic Logic to Alethic Modal Logic" [3]. In this paper, he proposed a reduction of obligation to violation. For example, the obligation to tell the truth means that a lie necessarily implies a violation. In general, and in its simplest form, something is obliged if and only if its absence necessarily leads to a violation.

The problems of these early approaches were illustrated in 1963 in a paper by Chisholm called "Contrary-to-duty imperatives and deontic logic" [16]. Consider a pregnant woman going to the hospital. The shortest way to go to the hospital is turning left, which obviously is what the driver is doing. However, there is a norm that it is forbidden to go to the left, so she is violating the obligation to go to the right. Now, the problem is due to two additional norms. One says that if she goes to the left she has to signal that she is going to the left, and one says that if she goes to the right she has to signal that she is going to the right. The problem here is how to explain that given that she is going to the left, she is obliged to signal that she is going to the left. This obligation cannot be explained by the basic distinction between good and bad, because the good thing here is to go to the right and signaling that she is going to the right at least from the perspective of traffic law.

Modern deontic logic started with a paper by Bengt Hansson in 1969, called "An Analysis of some Deontic Logics" [27]. In this paper he introduced a semantics based on a betterness relation for conditional obligations (it was axiomatized only six years later). With his paper he started modern deontic logic. The pregnant woman example can be represented by an ideal situation from the perspective of traffic law – in which the car goes to the right and signals that it will go to the right, but the situation in which the car goes to the left and signals that it will go to the left is better to the one in which the car goes to the left but signals that it will go to the right. As mentioned in the previous section, it is precisely the possibility of violation, that led Meyer and Wieringa to introduce the use of norms and deontic logic in computer science. Moreover, the formalism has become popular also in other areas of computer science too, such as non-monotonic logic and qualitative decision theory.

The seventh guideline says not to use the prehistory of deontic logic in the fifties and sixties of the previous century, but adapt to modern deontic logic as studied since the seventies. To say it crudely, Von Wright's and Anderson's systems have not been in the philosophical literature for forty years, so there seems little reason for computer scientists to return to these forgotten theories.

Guideline 7 *Use norms not only to distinguish right from wrong, but also to resolve dilemmas, and use norms not only describe violations, but in general to coordinate, organize, guide, regulate or control interaction among agents.*

Von Wright's system became known as the 'Old System', since he developed many modern systems too. Whereas the old system was based on monadic modal logic, the new systems were based on dyadic modal logics, just like Hansson's peference-based deontic logic. However, some people started to call the old system 'Standard Deontic Logic' or SDL, and this led to a lot of confusion. Some people in computer science, maybe due to the important role of standards in this research field, believed that a system called Standard Deontic Logic has to be a common reference for future explorations. To emphasize this misconception, let us consider some more examples. Remember that SDL sees norms just as being good and bad, or right and wrong. For example, it is right to obey your parents, it is wrong to hijack a plane, it is good to finish in time, and it is bad to write a computer virus. Though this is one way to look at norms, it is often not sufficient. Consider the following example. Suppose there is a plane hijacked by terrorists heading towards some high towers. There is a moral dilemma whether we may or should shoot down this plane. It is a dilemma, because if we shoot down the plane there will be a lot of innocent people killed, but if we don't shoot down the plane, then the plane will crash into these buildings. So it is a moral dilemma, and just thinking about right and wrong is not sufficient to solve it. People have been thinking about this kind of problems in ethics, and there are different theories. For example, a utilitarian theory says that you should minimize the damage. So what you should do is shoot down this plane, you may do it, you are obliged to do it, because if you don't do it, then the number of casualties will be higher than if you don't shoot it down. However, another ethical theory says that we may not shoot down the plane, because it is active involvement of ourselves, and if we do this, then we are responsible for killing the people in the plane. So it is forbidden to shoot down the plane. If we represent norms in computer systems, as we are now starting to do, then we can expect to find conflicts, and we therefore need to have a way to resolve these conflicts.

Anderson's reduction suggests that a norm is in the end just a description of violations. In the previous example of hijacking a plane, a norm says what counts as hijacking a plane, we call it a legal ontology, there is a norm telling us that it is a violation to hijack a plane, and there is a sanction associated when you do this. This is also very popular in computer science, but it is also insufficient. One indication of the problems related to this kind of reduction, is that people have not been able to give a reduction from Hansson's dyadic deontic logic to violations. Another conceptual problem is that violation is associated with norms and imperatives instead of obligations and prohibitions studied in deontic logic, an issue we discuss further below. But there are also practical problems. Consider for example the much discussed European Constitution. The question is, can we look at this text only as a set of descriptions of norm violations? There is an organizational structure, distribution of powers, there are several norms, there are several sanctions, and it seems, at least at first sight, that we can try to represent it as a set of norm violations. The problem with this constitution is that it has been rejected, we will never know what it really means. What we can do is look at the rules that are in force in the European Union, of which one says that the national deficit of a country should be below 3% of the national gross product. However, there were various countries who broke this norm, France and Germany in particular. However, the violation was not recognized, and the countries were not sanctioned.

The latter point is a little more subtle, because there are systems with violation predicates which are useful to reason about normative systems, namely diagnostic theories. Such theories have been developed in the eighties in artificial intelligence and computer science, and have been applied to a wide variety of domains, such as fault diagnosis of systems, or medical diagnosis. They can also be used to diagnose a court case, and determine whether someone is guilty or not (well, in principle, there are some more issues in legal reasoning which we will not consider here). However, as is well-known in ΔEON community, such a system is neither a deontic logic, nor a normative system. The main problem of such a formal system is that it does not deal easily with consequences of violations, so-called contrary-to-duty reasoning. For example, agent A should do α, and if he does not do so, then police agent B should punish him (a standard example in which norms regulate interaction between two agents). For a further discussion on this approach and its limitations, see [44].

The seventh guideline follows from the more recent literature on deontic logic, of which we have given a very sketchy overview in Table 4. In the beginning deontic logic was syntax based, and semantics came later. Modal semantics became very popular for some time, but during the past twenty years approaches based on non-monotonic logic and imperatives have become more and more popular. Nowadays, we can no longer say that deontic logic is a branch of modal logic. The use of possible worlds (Kripke) semantics is useful to distinguish good from bad, but less useful to represent dilemmas, or imperatives.

period	tradition	main issue
50s	monadic modal logic	relation O and P
60s	dyadic modal logic	relation O and facts, violations, sub-ideality and optimality, CTD
70s	temporal deontic logic	relation O and time
80s	action deontic logic	relation O and actions
90s	defeasible deontic logic	dilemmas, CTD
00s	imperatives, normative systems	Jorgensen's dilemma

Table 4. A schematic reconstruction of deontic logic

In particular, the seventh guideline follows from attempts during the past decade to base the semantics of deontic logic on imperatives. Deontic logic describes logical relations between obligations, prohibitions and permissions, but it is conditional on a normative system, which is typically left implicit. More precisely, there are two distinct philosophical traditions, the one of deontic logic discussed thus far, and another one of normative systems. The main challenge during the past ten years in deontic logic is how these two traditions can be merged. This story is explained in [25], which at this moment is the best introduction to current research in deontic logic. The most famous proponents of normative systems are Alchourrón and Bulygin [1], who argue in 1971 that a normative system should not be defined as a set of norms, as is commonly done, but in terms of consequences:

> "When a deductive correlation is such that the first sentence of the ordered pair is a case and the second is a solution, it will be called normative. If among the deductive correlations of the set α there is at least one normative correlation, we shall say that the set α has normative consequences. A system of sentences which has some normative consequences will be called a normative system." [1, p.55].

All the famous deontic logicians have discussed this subtle issue, often introducing new terminology. For example, Von Wright distinguished norms and normative propositions, and Alchourrón distinguished prescriptive and descriptive obligations.

Guideline 8 *Distinguish norms from obligations, prohibitions and permissions.*

As an example, consider the input/output logic framework introduced by Makinson and van der Torre [34]. The first input/output logic principle is that norms are not represented by propositional sentences, as in AGM framework for theory change [2], or as modal formulas, as in deontic logic, but as pairs of formulas of an arbitrary logic. The pair of propositional formulas represents a rule, and the two propositional formulas are called the antecedent and consequent of the rule. The second principle of the input/output logic framework is that the primary role of norms in a normative system is the derivation of obligations and prohibitions. Which obligations and prohibitions can be derived from a normative system depends on the factual situation, which we call the *context* or *input* and represent by a propositional formula. The function that associates with each context the set of obligations describes the meaning of the normative system, because it is a kind of 'operational semantics' of the normative system. An input/output operation $out : (2^{L \times L}) \times L \to 2^L$ is a function from the set of normative systems and contexts, to a set of sentences of L. We say that x is obligatory in normative system N and context a if $x \in out(N, a)$. The simplest input/output logic defined by Makinson and van der Torre is so-called simple-minded output. x is in the simple-minded output of N in context a, written as $x \in out_1(N, a)$, if there is a set of norms $(a_1, x_1), \ldots, (a_n, x_n) \in N$ such that $a_i \in Cn(a)$ and $x \in Cn(x_1 \wedge \ldots \wedge x_n)$, where $Cn(S)$ is the consequence set of S in L. Such an operational semantics can be axiomatized as follows. $out_1(N)$ is the minimal set that contains $N \cup \{(\top, \top)\}$, is closed under replacement of logical equivalents in antecedent and consequent, and the following proof rules strengthening of the input SI, weakening of the output WO, and conjunction rule AND.

$$\frac{(a,x)}{(a \wedge b, x)} SI \qquad \frac{(a, x \wedge y)}{(a,x)} WO \qquad \frac{(a,x),(a,y)}{(a, x \wedge y)} AND$$

Ten philosophical problems on deontic logic are given by Hansen *et al.* [26] and listed in Table 5. Of this list, we observe that constitutive norms and intermediate concepts are often seen as the same problem (though constitutive norms can be used also for other problems than intermediate concepts), and that there are other problems not listed in this paper, such as the equivalence of normative systems, or redundancy of a norm in a normative system [12, 5].

1. How can deontic logic be reconstructed in accord with the philosophical position that norms are neither true nor false?
2. When is a set of norms to be termed 'coherent'?
3. How can deontic logic accommodate possible conflicts of norms? How can the resolution of apparent conflicts be semantically modeled?
4. How do we reason with contrary-to-duty obligations which are in force only in case of norm violations?
5. How to define dyadic deontic operators with regard to given sets of norms and facts?
6. How to distinguish various kinds of permissions and relate them to obligations?
7. How can meaning postulates and intermediate terms be modeled in semantics for deontic logic reasoning?
8. How to define counts-as conditionals and relate them to obligations and permissions?
9. How to revise a set of regulations or obligations? Does belief revision offer a satisfactory framework for norm revision? Can the belief merging framework deal with the problem of merging sets of norms?

Table 5. Ten philosophical problems. "We argue that norms, not ideality, should take the central position in deontic semantics, and that a semantics that represents norms, as input/output logic does, provides helpful tools for analyzing, clarifying and solving the problems of deontic logic." [26]

3.2 Methodology

Not surprisingly for such a highly simplified theory like Von Wright's Old System, also know as SDL, there are many features of actual normative reasoning that SDL does not capture. Notorious are the so-called 'paradoxes of deontic logic', which are usually dismissed as consequences of the simplifications of SDL. For example, Ross's paradox [38], the counterintuitive derivation of "you ought to mail or burn the letter" from "you ought to mail the letter", is typically viewed as a side effect of the interpretation of 'or' in natural language.

Guideline 9 *Don't motivate your new theory by toy "paradoxical" examples, but use the deontic paradoxes to illustrate basic properties of your system.*

Computer scientists are usually surprised when they read the philosophical literature, because the posed problems seem to have a trivial solution. For example, the most famous deontic paradox of all, is often posed as the problem to give a consistent representation such that none of the sentences can be derived from the others:

1. A certain man should go to the assistance of his neighbors,
2. If he goes, he should tell them he is coming
3. If he does not go, he should not tell them that he is coming
4. He does not go.

In SDL the set $\{Oa, O(a \rightarrow t), \neg a \rightarrow O(\neg t), \neg a\}$ is inconsistent, and in $\{Oa, a \rightarrow O(t), \neg a \rightarrow O(\neg t), \neg a\}$ the sentences are not logically independent. However, this problem is trivially solved by replacing the material implication by a strict implication, or a relevant implication, or a defeasible implication. This has been known since the early days of deontic logic, as may be expected since both deontic logic and conditional logic were major branches of philosophical logic. In general, any paradoxical consequence can be solved by simply weakening the logic (e.g., solve Ross' paradox by replacing standard deontic logic by a non-normal modal logic). The misconception is simply due to the fact that the deontic paradoxes do not work the same way as experiments in engineering or the sciences. They are just used to illustrate the formal system, not to guide research in the area. A similar phenomena and misconception has been present in the field of defeasible reasoning and deontic logic, where the use of the infamous Tweety example has been criticized for similar reasons. The reason why contrary-to-duty paradoxes have been discussed for fifty years in deontic logic is that a lot of normative reasoning is directly or indirectly related to violations, just like in defeasible reasoning a lot of reasoning is directly or indirectly related to exceptions.

3.3 From philosophy to computer science

Most of the confusions in deontic logic are due to the abstract nature of the formal systems. In areas of computer science like multiagent systems or knowledge representation we need to be more detailed, and most of the problems then disappear.

Guideline 10 *Regulative norms should not be considered by themselves, but in relation to permissive norms, constitutive norms, procedural norms, agents, roles, groups, societies, rights, duties, obligations, time, beliefs, desires, intentions, goals, roles, and other kinds of norms and other social-cognitive computer science concepts.*

Regulative norms specify the ideal and varying degrees of sub-ideal behavior of a system by means of obligations, prohibitions and permissions. Constitutive norms are based on the notion that "X counts-as Y in context C" and are used to support regulative norms by introducing institutional facts in the representation of legal reality. The notion of counts-as introduced by Searle [40] has been interpreted in deontic logic in different ways and it seems to refer to different albeit related phenomena [24]. Substantive norms define the legal relationships of people with other people and the state in terms of regulative and constitutive norms, where regulative norms are obligations, prohibitions and permissions, and constitutive norms state what counts as institutional facts in a normative system. Procedural norms are instrumental norms, addressed to agents playing roles in the normative system, which aim at achieving the social order specified in terms of substantive norms [7].

4 Summary

Next generation normative multiagent systems contain general and domain independent norms by combining three existing representations of normative multiagent systems. First, theories of normative systems and deontic logic, the logic of obligations and permissions, for the explicit representation of norms as rules, the application of such rules, contrary-to-duty reasoning and the relation to permissions. Second, agent architecture for software engineering of agents and a model of normative decision making. Third, a game-theoretic approach for model of interaction explaining the relation among social norms and obligations, relating regulative norms to constitutive norms, the evolution of normative systems, and much more. In this paper, we introduce and discuss ten guidelines for the development of normative multiagent systems.

1. Motivate which definition of normative multiagent system is used.
2. Make explicit why norms are a kind of (soft) constraints deserving special analysis.
3. Explain why and how norms can be changed at runtime.
4. Discuss the use and role of norms as a mechanism in a game-theoretic setting.
5. Clarify the role of norms in the multiagent system.
6. Relate the notion of "norm" to the legal, social, or moral literature.
7. Use norms not only to distinguish right from wrong, but also to resolve dilemmas, and use norms not only describe violations, but in general to coordinate, organize, guide, regulate or control interaction among agents.
8. Distinguish norms from obligations, prohibitions and permissions.
9. Use the deontic paradoxes only to illustrate the normative multiagent system.
10. Consider regulative norms in relation to other kinds of norms and concepts.

Table 6. Ten guidelines for the development of normative multiagent systems

The use of norms and normative systems in computer science are examples of the use of social concepts in computer science, which is now so well-established that the original meaning of some of these concepts in the social sciences is sometimes forgotten. For example, the original meaning of a "service" in business economics is rarely considered by computer scientists working on service oriented architectures or web services, and likewise for service level agreements and contracts, or quality of service. some social concepts have various new meanings. For example, before its use in service level agreements, the notion of "contract" was introduced in software engineering in Meyer's design by contract, a well known software design methodology that views software construction as based on contracts between clients (callers) and suppliers (routines), assertions, that has been developed in the context of object oriented and the basis of the programming language Eiffel. “Coordination” is emerging as an interdisciplinary concept to deal with the complexity of compositionality and interaction, and has been used from coordination languages in software engineering to a general interaction concept in multiagent systems. In the context of information security and access control “roles” became popular, with the popularity of eBay, the social concepts of “trust” and “reputation” have become popular, and with the emergence of social interaction sites like FaceBook or Second Life, new social concepts like societies, coalitions, organizations, institutions, norms, power, and trust are emerging [15]. In multiagent systems, social ability as the interaction with other agents and co-operation is one of the three meanings of flexibility in flexible autonomous action in Wooldridge and Jennings’ weak notion of agency [46]; the other two are reactivity as interaction with the environment, and proactiveness as taking the initiative.

The main open question is whether “norms” could (or should) play a similar role in computer science like “service”, “contract” or “trust”? One suggestion comes from human computer interaction. Since the use of norms is a key element of human social intelligence, norms may be essential too for artificial agents that collaborate with humans, or that are to display behavior comparable to human intelligent behavior. By integrating norms and individual intelligence normative multiagent systems provide a promising model for human and artificial agent cooperation and co-ordination, group decision making, multiagent organizations, regulated societies, electronic institutions, secure multiagent systems, and so on. Another suggestion comes from the shared interest of multiagent system research and sociology in the relation between micro-level agent behaviour and macro-level system effects. Norms are thought to ensure efficiency at the level of the multiagent system whilst respecting individual autonomy. However, all these and other suggestions bring circumstantial evidence at best. We have to build more flexible normative multiagent systems, and test them in practice, before we know where they can be used best.

For further reading on the use of normative systems in computer science, we recommend the proceedings of the ΔEON conferences and the normative multiagent systems workshops. The abstracts of all papers that appeared at DLCS conferences can be searched on the deontic logic website:

`http:\\deonticlogic.org`

References

1. C. Alchourrón and E. Bulygin. *Normative Systems.* Springer, Wien, 1971.
2. C. Alchourrón, P. Gärdenfors, and D. Makinson. On the logic of theory change. *Journal of Symbolic Logic*, 50(2):510–530, 1985.

3. A. Anderson. A reduction of deontic logic to alethic modal logic. *Mind*, 67:100–103, 1958.
4. M. Anderson and S. Leigh Anderson. Machine ethics: Creating an ethical intelligent agent. *AI Magazine*, 28(4):15–26, 2007.
5. G. Boella, J. Broersen, and L. van der Torre. Reasoning about constitutive norms, counts-as conditionals, institutions, deadlines and violations. In *Intelligent Agents and Multi-Agent Systems, 11th Pacific Rim International Conference on Multi-Agents, PRIMA 2008, Hanoi, Vietnam, December 15-16, 2008. Proceedings*, volume 5357 of *Lecture Notes in Computer Science*, pages 86–97. Springer, 2008.
6. G. Boella and L. van der Torre. The ontological properties of social roles in multi-agent systems: Definitional dependence, powers and roles playing roles. *Artificial Intelligence and Law Journal (AILaw)*, 2007.
7. G. Boella and L. van der Torre. Substantive and procedural norms in normative multiagent systems. *Journal of Applied Logic*, 6(2):152–171, 2008.
8. G. Boella, L. van der Torre, and H. Verhagen. Introduction to normative multiagent systems. *Computation and Mathematical Organizational Theory, special issue on normative multiagent systems*, 12(2-3):71–79, 2006.
9. G. Boella, L. van der Torre, and H. Verhagen. Normative multi-agent systems. In *Internationales Begegnungs und Porschungszentrum fur Informatik (IBFI)*, 2007.
10. G. Boella, H. Verhagen, and L. van der Torre. Introduction to the special issue on normative multiagent systems. *Journal of Autonomous Agents and Multi Agent Systems*, 17(1):1–10, 2008.
11. M.E. Bratman. *Intentions, plans, and practical reason*. Harvard University Press, Harvard (Massachusetts), 1987.
12. J. Broersen and L. van der Torre. Reasoning about norms, obligations, time and agents. In *Intelligent Agents and Multi-Agent Systems, 10th Pacific Rim International Conference on Multi-Agents, PRIMA 2007, Proceedings*, Lecture Notes in Computer Science. Springer, 2007.
13. E. Bulygin. Permissive norms and normative systems. In A. Martino and F. Socci Natali, editors, *Automated Analysis of Legal Texts*, pages 211–218. Publishing Company, Amsterdam, 1986.
14. C. Castelfranchi. Modeling social action for AI agents. *Artificial Intelligence*, 103(1-2):157–182, 1998.
15. C. Castelfranchi. The micro-macro constitution of power. *Protosociology*, 18:208–269, 2003.
16. R.M. Chisholm. Contrary-to-duty imperatives and deontic logic. *Analysis*, 24:33–36, 1963.
17. R. Conte, C. Castelfranchi, and F. Dignum. Autonomous norm-acceptance. In *Intelligent Agents V (ATAL98)*, LNAI 1555, pages 319–333. Springer, 1999.
18. A. Giddens. *The Constitution of Society*. University of California Press, 1984.
19. P. J. Gmytrasiewicz and E. H. Durfee. Formalization of recursive modeling. In *Procs. of ICMAS'95*, pages 125–132, Cambridge (MA), 1995. AAAI/MIT Press.
20. U. Gneezy and A. Rustichini. A fine is a price. *The Journal of Legal Studies*, 29(1):1–18, 2000.
21. L. Goble and J.J. Ch. Meyer, editors. *Deontic Logic and Artificial Normative Systems, 8th International Workshop on Deontic Logic in Computer Science, DEON 2006, Utrecht, The Netherlands, July 12-14, 2006, Proceedings*, volume 4048 of *Lecture Notes in Computer Science*. Springer, 2006.
22. L. Goble and J.J. Ch. Meyer. Revised versions of papers presented in the proceeding of the eighth international workshop on deontic logic in computer science (DEON06). *Journal of Applied Logic*, 6(2), 2008.

23. E. Goffman. *The Presentation of Self in Everyday Life*. Doubleday, 1959.
24. D. Grossi, J.-J.Ch. Meyer, and F. Dignum. Counts-as: Classification or constitution? an answer using modal logic. In *Procs. of Deontic Logic and Artificial Normative Systems, 8th International Workshop on Deontic Logic in Computer Science, (ΔEON'06)*, volume 4048 of *LNCS*, pages 115–130, Berlin, 2006. Springer.
25. J. Hansen. *Imperatives and Deontic Logic*. PhD thesis, University of Leipzig, 2008.
26. J. Hansen, G. Pigozzi, and L. van der Torre. Ten philosophical problems in deontic logic. In G. Boella, L. van der Torre, and H. Verhagen, editors, *Normative Multi-agent Systems*, volume 07122 of *Dagstuhl Seminar Proceedings*. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany, 2007.
27. B. Hansson. An analysis of some deontic logics. *Noûs*, 3:373–398, 1969.
28. M. Hollis. *Trust within reason*. Cambridge University Press, Cambridge, 1998.
29. Steven D. Levitt and Stephen J. Dubner. *Freakonomics : A Rogue Economist Explores the Hidden Side of Everything*. William Morrow, New York, May 2005.
30. D. Lewis. A problem about permission. In E. Saarinen, editor, *Essays in Honour of Jaakko Hintikka*, pages 163–175. D. Reidel, Dordrecht, 1979.
31. A. Lomuscio and D. Nute, editors. *Deontic Logic in Computer Science, 7th International Workshop on Deontic Logic in Computer Science, DEON 2004, Madeira, Portugal, May 26-28, 2004. Proceedings*, volume 3065 of *Lecture Notes in Computer Science*. Springer, 2004.
32. A. Lomuscio and D. Nute. Revised versions of papers presented in the proceeding of the seventh international workshop on deontic logic in computer science (DEON04). *Journal of Applied Logic*, 3(3-4), 2005.
33. M. Luck, P. McBurney, and C. Preist. *Agent Technology: Enabling Next Generation Computing (A Roadmap for Agent Based Computing)*. AgentLink, 2003.
34. D. Makinson and L. van der Torre. Input-output logics. *Journal of Philosophical Logic*, 29(4):383–408, 2000.
35. Merriam-Webster. *Online dictionary http://www.merriam-webster.com/*. Merriam-Webster.
36. J.-J. Meyer and R. Wieringa. *Deontic Logic in Computer Science: Normative System Specification*. John Wiley & Sons, Chichester, England, 1993.
37. A. Newell. The knowledge level. *Artificial Intelligence*, 18:87–127, 1982.
38. A. Ross. Imperatives and logic. *Theoria*, 7:53–71, 1941. Reprinted in *Philosophy of Science* **11**:30–46, 1944.
39. D.W.P. Ruiter. A basic classification of legal institutions. *Ratio Juris*, 10(4):357–371, 1997.
40. J.R. Searle. *The Construction of Social Reality*. The Free Press, New York, 1995.
41. Y. Shoham and M. Tennenholtz. On social laws for artificial agent societies: Off-line design. *Artificial Intelligence*, 73(1-2):231–252, 1995.
42. A. Strauss. *Negotiations: Varieties, Contexts, Processes and Social Order*. San Francisco, Jossey-Bass, 1978.
43. R. van der Meyden and L. van der Torre, editors. *Deontic Logic in Computer Science, 9th International Conference on Deontic Logic in Computer Science, DEON 2008, Luxembourg, July 16-18, 2008, Proceedings*, LNCS, Berlin, in press. Springer.
44. L. van der Torre and Y. Tan. Diagnosis and decision making in normative reasoning. *Artificial Intelligence and Law*, 7(1):51–67, 1999.
45. G. H. von Wright. Deontic logic. *Mind*, 60:1–15, 1951.
46. M. J. Wooldridge and N. R. Jennings. Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10(2):115–152, 1995.