F. Dignum. Autonomous agents and social norms. In R. Falcone and R. Conte, editors, *ICMAS-96 Workshop on Norms, Obligations and Conventions*, pages 56--71, Kyoto, 1996.

Autonomous Agents and Social Norms

F.Dignum
Fac. of Maths. & Comp. Sc., Eindhoven University of Technology
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
e-mail: dignum@win.tue.nl

Abstract

In this paper we present concepts and their relations that are necessary for modeling autonomous agents in an environment that is governed by some (social) norms. We divide the norms over three levels: the private level the contract level and the convention level. We show how deontic logic can be used to model the concepts and how the theory of speech acts can be used to model the generation of (some of) the norms. Finally we give some idea about an agent architecture incorporating the social norms based on a BDI framework.

1 Introduction

In the area of Multi-Agent Systems much research is devoted to the coordination of the agents. Many papers have been written about protocols (like Contract-Net) that allow agents to negotiate and cooperate (e.g. [13, 5]). Most of the cooperation between agents is based on the assumption that they have some joint goal or intention. Such a joint goal enforces some type of cooperative behaviour on all agents (see e.g. [4, 10, 20]). The conventions according to which the agents coordinate their behaviour is hard-wired into the protocols that the agents use to react to the behaviour (cq. messages) of other agents.

This raises several issues. The first issue is that, although agents are said to be autonomous, they always react in a predictable way to each message. Namely their response will follow the protocol that was built-in. The question then arises how autonomous these agents actually are. It seems that they react always in standard ways to some stimulus from other agents, that can therefore determine their behaviour.

Besides autonomy, an important characteristic of agents is that they can react to a changing environment. However, if the protocols that they use to react to (at least some part of) the environment are fixed, they have no ways to respond to changes. For instance, if an agent notices that another agent is cheating it cannot switch to another protocol to protect itself. (At least this is not very common). In general it is difficult (if not impossible) for agents to react to violations of the conventions by other agents.

Related to this issue is the fact that if the conventions are hard-wired into the agent's protocols it cannot decide to violate the conventions. There might be circumstances in which the agent violates a convention in order to adhere to a private goal that it considers

to be more important (more profitable). For instance, delete a file that contains a virus, while the agent should not delete files.

In this paper we will argue that deontic logic can be used to model the norms according to which agents interact with each other. Deontic logic gives the opportunity to explicitly describe the norms that can be used to implement the interactions between agents. Also it can be used to model violations of these norms and possible reactions on these violations.

We distinguish three levels on which the social behaviour of an agent is determined. First is the private level. On this level the agent makes private judgements between different obligations and/or goals and determines the actions it will take. The choice of actions or goal to be pursued are usually indicated with the intentions and commitment of the agent. This level is described in section 5.

The next level is the contract level, which is described in the section 3. On this level the actual obligations that exist between agents are described. These obligations create a certain dependency between the agents. An important part of this level, however, is the description of repercussions in case of violations. Also on this level we can describe responsibility, authorization and power relations. between agents.

The highest level is the convention level. This level describes the conventions according to which the agents should coordinate their actions. These conventions can be very diverse. For instance, "any request from another agent should get an answer (either positive or negative)". But also "An agent should be cooperative (if possible)". The convention level is described in section 4.

Before we describe the different levels of norms and their relations, we will describe an agent architecture, based on [23], in which the norms are implemented in different knowledge bases. The private part of this agent architecture can be seen as a variant of the classical BDI architecture [17]. It also resembles very closely the agent architecture of the ADEPT system described in [15].

Finally, in section 6, we give some first conclusions on the basis of our framework.

2 Agent Architecture

In figure 1 we show how the different levels of social behaviour that we define can be incorporated into an agent architecture.

The agent consists of an active part, the interpreter, and several knowledge bases. In the interpreter we distinguish a communication manager, a contract manager, a task manager and a service manager. The communication manager has knowledge of the communication protocols and the state of the agent. On the basis of this knowledge it can send and process messages. Although it is an important aspect of the agent it is not relevant for this paper.

The service execution manager takes care that the services that an agent provides for other agents or itself get executed. It handles some exceptions and is used by the task and contract manager whenever services of other agents are needed. The service manager knows about the services the agent itself can provide and about services of agents it dealt with

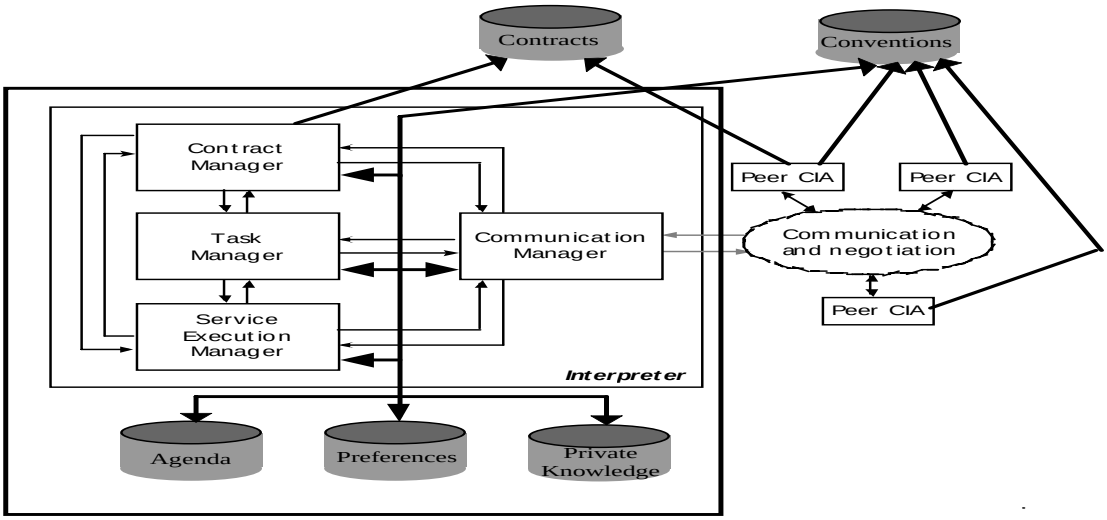

Figure 1: agent architecture

in the past.
The contract manager monitors the fulfillment of the contracts for which the agent is responsible. It takes appropriate action whenever a contract is violated. The contract manager also establishes new contracts. It provides the knowledge for the negotiation that is done through the communication manager. It is clear that the contract manager takes care adhering to all the norms that belong to the contract level. The contracts themselves are stored in a database that is accessible by all agents involved in the contract.
The most important part of the agent concerning normative behaviour is the task manager. The task manager determines which tasks (actions) will be performed by the agent and in which order. It also handles failures and tries to find alternatives. The task manager uses the agenda (Intentions), the private preferences of the agent (Desires) and the knowledge about the world (Believes) to determine the next task for the agent.
The conventions are implemented in this architecture as a central database that can be accessed by all agents. As we will later argue it might be the case that the conventions are seen as a contract between a special agent and all other agents.

The different concepts that we define in the next sections actually describe the contents of the different data (or knowledge) bases in the above architecture. Only in section 5 we indicate how the concepts can be used by the task manager to effectuate normative behaviour.
The task manager has to combine most concepts of the different layers to determine which step to take next. Therefore we think it is essential that all these concepts are formally defined and expressed in a uniform way. This is done in a multi-modal logic. Although we do use the notation of this logic (sometimes) to describe the concepts, we do not formally introduce the complete logic. At some places we indicate some of the more important properties of the logical operators that describe the concepts. For a more detailed description we refer to our other work, because we feel it would distract to much from the main points

that we want to make in this paper. The main purpose of using the logic as a semantics of the concepts is that it shows that the concepts can be formally defined in a uniform way. However, we feel that the concepts themselves are also usefull if any other formalism is used to describe social behaviour of agents.

3 Contracts

We will start with the contract level, because the deontic concepts that we use to model the social norms are best explained at this level. It will be shown that the deontic concepts needed to model the other levels can be seen as special cases of the ones defined at the contract level.

In our view contracts are centred around obligations and authorizations. For each obligation and authorization we indicate how it arises, how it is fulfilled (or expires) and what happens if it is violated. Not only legal contracts but also cooperation and informal agreements between agents can be described in this way. The contract describes the type of relation that exists between the agents and their mutual expectations of the behaviour of the other agent. In [24] we describe more fully how the contracts can be implemented using a formal language CoLa.

3.1 Directed obligations

The central notion that is used to model norms on the contract level is the *directed obligation* (see e.g. [9, 19]). It is defined as follows:

> $O_{ij}(p)$ means that agent i is obliged towards agent j (the counterparty) that p holds.
> $O_{ij}(\alpha)$ means that agent i is obliged towards agent j (the counterparty) that α is performed.

Note that we distinguish between obligations about situations and actions. The distinction has a practical reason. Actions that are obliged can be simply put on the "agenda", while for situations that are obliged a plan has to be devised to reach them.
They can formally be reduced to only obligations on situations, as shown in [8], by considering the obligation to perform α to be the same as the fact that α should have been performed at some point in the future. Usually obligations on actions carry a time aspect indicating that the action should be performed before a certain deadline. We abstract from this feature here, but have described it fully in [6].

For agent i the directed obligation means that it should perform some action to fulfill the obligation. Agent j has a conditional power or *authorization* to "repair" the situation in case i does not fulfill its obligation. This means that j can in those cases demand further actions from i, cancel some of its own obligations towards i or perform some repair action himself.

The directed obligations O_{ij} specify a loose coordination between two agents. It creates incentives for agent i to perform some action or reach a goal. For agent j it creates expectations about the behaviour of agent i. However, both agents are still autonomous.

Agent i might decide not to perform the actions it is obliged to. The use of deontic logic gives the opportunity to specify explicitly what should happen in these cases of violation of the obligations. We illustrate this with a small part of a contract between an airline and a passenger. We first give the CoLa description and then the formal logic interpretation:

```
Obl(a,p,transport_passenger(a,p))
in:     flight_reservation(p,a)
goal:   transport_passenger(a,p)
exit:   cancel(p,ticket) => Obl(p,a,pay_costs)
        cancel(a,flight) => Obl(a,p,pay_costs)
```

The above means that the obligation for an airline to transport a passenger arises from a flight reservation. It can be fulfilled by transporting the passenger and it is "violated" if the passenger cancels the flight or the airline cancels the flight.
In logic this is described as follows:

$$[flight_reservation(a,p)]O_{ap}(transport-passenger)$$
$$O_{ap}(transport-passenger) \rightarrow [cancel(p,ticket)]O_{pa}(pay-costs)$$
$$O_{ap}(transport-passenger) \rightarrow [cancel(a,flight)]O_{ap}(pay-costs)$$

where $[\alpha]\phi$ means that after the performance of α the formula ϕ holds. We assume here that the cancelations imply the non-performance of the obligation.

3.2 Authorization

A second concept that is very important in the relation between agents is that of authorization. If an agent has the authorization to perform some action it has some basis on which to justify it. For actions that have a physical effect it can be equated with permission. If a person is authorized to change a database then he is not in violation after he actually did it.
For actions that have an abstract effect (like speech acts) authorization encompasses permission. It is not only permitted to perform the action but the abstract effect of the action is also ensured. The most important example is that if agent i is authorized to direct agent j to perform some action then, after the command, j has the actual obligation to perform the action! E.g. if i is authorized to demand payment from j then j is obliged to pay after the demand to do so. This is not the case if i is not authorized!

As can be seen from the above, the authorization of agents is the basic factor in the formation of coordination between agents. Agents with many authorizations can dominate other agents and have a lot of power to achieve their goals. Authorizations can be generated in different ways. First they can be build in by the programmers. However, this can only be done if all agents are made by the same standards. If two agents communicate from different systems they will probably not recognize each others authorizations.
A second way to establish authorizations that is related to the first one, is the linking of authorizations to the functions or roles that agents have. E.g. a consumer agent is

authorized to request prices of products. An agent that explicitly coordinates several other agents is authorized to command them to perform some task, etc.
The third way authorizations are generated is through implicit effects of actions. The effect of accepting a delivery of a product implicitly authorizes the producer to demand payment. These implicit effects of actions are defined on the convention level.
The last way to generate authorizations is by explicit creation by the agents. One agent can explicitly authorize another agent to perform some actions. We will come back to this form later on.

At the moment we model the authorization with a special predicate with two arguments: the agent and the action it is authorized to perform. This is a very simplified way to capture the authorization, which in its full form should also contain elements of time and context. We leave this for future work.

We finish again with a small example of an authorization in a contract:

```
auth(a,direct(a,p,pay_ticket(p)))
in:     flight_reservation(p,a)
goal:   pay_ticket(p)
```

This means that the airline is authorized to order the passenger to pay after a reservation has been made. The authorization finishes after the payment by the passenger. In logic this is described as:

$$[flight - reservation(p,a)]auth(a, DIRECT(a,p,pay - ticket(p)))$$
$$[pay - ticket(p)]\neg auth(a, DIRECT(a,p,pay - ticket(p)))$$

In the above formalization $DIRECT(a,p,pay - ticket(p))$ is a speech act from the airline a to the passenger p with illocution $DIRECT$, in which the airline orders (directs) the passenger to perform the action $pay - ticket(p)$.

3.3 Responsibility

A third concept that plays an important role on the contract level is responsibility. Although we think that real responsibility is only applicable to humans it is possible to translate the concept to the domain of agents. In the domain of agents the concept of responsibility is especially important when we consider the breakdown of the system. Whenever an action that had to be performed did not actually take place then it is the task of the responsible agent to "repair" the situation. This is of particular interest when an agent is obliged to have an action performed but has another agent actually performing it. A very common practice in the building and transport industry. In this case the agent performing the task might not be responsible if the task cannot be completed successfully. Within our framework we will identify the concept of responsibility with the obligation. I.e. if an agent has the obligation to perform an action it is also responsible for that action. The obligation does not have to be an obligation towards another agent but can also be an obligation towards itself. This can be used to model the responsibilities from an agent stemming from the function or role of the agent.

3.4 Practical considerations

Although all obligations on the contract level can be modeled using the same logical concept of directed obligation, some distinctions should be made for practical purposes.
A first distinction should be made between directed obligations that are concrete and those that are abstract. Concrete obligations define one particular situation that has to be reached or an action that has to be performed. For instance, "the customer has to pay within 3 weeks" or "the passenger should be at the airport one hour before the flight leaves". This type of obligations can be modeled directly in a deontic logic with time as defined in [6]. These formulas can immediately be used by the agent to form goals and/or plans to fulfill the obligation.

The abstract obligations define a (vague) class of situations or actions. For instance, "agent i should cooperate with agent j". Depending on the situation this can be done in different ways. The abstract obligations first need an interpretation to define their meaning in a concrete situation. These interpretation rules are defined on the convention level.

Another practical distinction that should be made is that between one-time and general situations and actions. The situations that should be reached only once can be seen as a goal. A plan can be made to reach the goal and once it is reached the obligation is fulfilled. A good example is the payment of an order. This should be done once only!
There are also obligations that are not fulfilled if a certain situation is reached once. For instance, "agent i should give agent j all information about agents that it encounters". This obligation will not disappear through an act of agent i. Every time agent i encounters information about agents it should again give it to agent j as well.
Actually the last type of obligations can never be fulfilled, but they can be violated. In the contract description using CoLa this distinction can be made very simply by NOT giving a goal state for these obligations. Logically the two types of obligations are distinguished by preceding the general obligations with a temporal "always" operator.

3.5 Generating obligations and authorizations

Obligations can be formed either through the implicit effect of an action, which is defined on the convention level or through some special type of messages.
Obligations can arise implicitly from actions like "accept order". The acceptance of the order can imply the obligation to deliver the product. This implicit generation of an obligation stems from the fact that there already exists a (standard) contract between the agents in which these dependencies are included. These contracts take a place between the actual contract level and the convention level. There usually is a set of standard contracts that agents can use for their coordination. A contract becomes actual for two agents when the two agents agree to use that contract. The contract consists mainly of conditional obligations that become actual in a certain situation or after a certain event takes place. In terms of CoLa that is after the triggering (or "in") event of an obligation has occurred.

Obligations can also be created through sending messages, i.e. communication between agents. The communication between agents is modeled on the basis of speech act theory as described in [21, 22]. They describe five categories of speech acts with their own

characteristics. In [7] we give a formal semantics for the speech act types and show how they can be used to create obligations, believes and facts. At this place we only want to state that obligations can arise from a commitment of one agent to another. E.g. I promise to deliver the goods on Friday leads to the obligation to do so. Formally:

$$[COMMIT(i, j, deliver)]O_{ij}(deliver)$$

An obligation can also arise through an authorized command. E.g. a demand for payment after the goods are delivered leads to an obligation to do so. Formally:

$$auth(i, DIRECT(i, j, pay)) \rightarrow [DIRECT(i, j, pay)]O_{ji}(pay)$$

The authorizations are created also by convention or through the special role or function of the agent or through special "authorization" messages. If the agent fulfills certain requirements it can authorize another agent to perform some action. This is very important because most actions only achieve their desired effect if the actor is authorized. E.g. agent i could agree with agent j to always deliver his goods on request by giving him an authorization to ask for delivery. Formally:

$$[AUT(i, j, DIRECT(j, i, deliver))]auth(j, DIRECT(j, i, deliver)) \wedge$$
$$auth(j, DIRECT(j, i, deliver)) \rightarrow [DIRECT(j, i, deliver)]O_{ij}(deliver)$$

Of course an agent cannot authorize another agent to perform every action. The actions that an agent can authorize another agent to perform depend on the role of the agent. With each role there are a number of standard authorizations. Besides these authorizations each agent can authorize any other agent to perform a directive towards itself. For more (formal) details about these authorizations see [7].

4 Conventions

The level of conventions between agents can be compared with the *prima facie* obligations that arise from the law. Prima facie obligations hold under normal circumstances, becoming actual unless some other moral consideration intervenes ([18, 1]). They provide a kind of "moral background" against which people (and agents) interact.

We distinguish two types of conventions, concrete conventions (or interpretation rules) and abstract conventions (or prima facie norms).

4.1 Interpretation rules

There are two types of concrete conventions. The first type are interpretation rules indicating how evaluative terms should be interpreted in certain situations. Evaluative terms are terms like "reasonable", "good", "cheap", etc. The conventions give a kind of conditional definition of these terms. E.g. Suppose that agent i should deliver computers to agent j against reasonable market prices. The convention might state that (under normal circumstances) reasonable market prices are not more than 10% above the lowest price on the market.

The second type of concrete conventions are used to describe that certain actions will have a certain implicit (deontic) effect. E.g. If agent i orders a product from agent j then he implicitly authorizes j to demand payment upon delivery of the goods.

The above two examples can be described in a formal language (ConLa):

```
INTERPRETATION RULES

Definitions
reasonable_price(p) DEF NOT(p > lowest_price+10%)

Implicit effects
order_product(i,j) => (deliver(j,product) => auth(j,direct(j,i,pay)))
```

Logically these conventions are (conditional) implications that function as axioms for the system in which the agents function.
The first type of conventions are written as logical equivalences:

$$Reasonable(p) \leftrightarrow (\exists p' \forall p" p' < p" \rightarrow p < p' + 10\%)$$

The second type of conventions can be modeled using dynamic logic:

$$[Order(i,j,p)][Deliver(j,i,p)]Auth(j, DIRECT(j,i,pay))$$

This means that always after $Order(i,j,p)$ followed by $Deliver(j,i,p)$,j gets the authorization to direct i to pay him.

4.2 Prima facie norms

The abstract conventions are probably the best example of pure deontic sentences describing general social norms and values (also called prima facie norms). There are three types of normative positions. The prohibition, the permission and the obligation.
In deontic logic we can model the three types of normative positions with three modal operators, the F for prohibition, the P for permission and the O for obligation. Unfortunately there is not one standard semantics for these operators. We will use a semantics in which the operators can be applied over both actions and propositions, because we think it is important to have normative positions over both.

The prohibitions function as limitations on the behavior of agents. E.g. "Agents cannot copy information without authorization (of the owner of that information)". This could be described in deontic logic as

$$\forall i \neg auth(i, copy) \rightarrow F_i(copy)$$

Crucial in this case is that there is still the possibility that the agent copies information without authorization. This will lead to a situation of violation of the norm, but not an

inconsistent state. It is also still possible to specify what should be done in this case. For instance let the agent pay a fine:

$$F_i(copy) \rightarrow [copy]O_i(pay - fine)$$

If the above norm would be modeled (as done in many systems) as:

$$\neg auth(i, copy) \rightarrow \neg DO(copy)$$

where $DO(copy)$ stands for the fact that the next action is *copy*. In this case if the agent still copies the information we get into an inconsistent state. $(DO(copy) \wedge \neg DO(copy))$
It shows that the use of deontic concepts preserves the autonomy of the agents.

The permission operator is almost only used to indicate exceptions to a general rule or in cases of uncertainty. E.g. "persons are permitted to kill in self defense", which is an exception to the general rule that persons cannot kill other persons".

The obligations are descriptions of an ideal situation. For instance, "Agents should behave cooperatively". These types of norms cannot be transformed into goals because they are not situations that can be actually reached. However, they can be used to evaluate different possible actions and choose the most appropriate. So, in contrast to the contract level the obligations on the convention level do not (usually) give rise to actions, but are only used to choose between alternative courses of action.

The above examples are written as follows in the formal language ConLa:

```
PRIMA FACIE NORMS

forb(i,copy(i,information))
in:     NOT auth(i,copy(i,information))
exit:   copy(i,information) => obl(i,pay_fine(i))

perm(i,kill(i))
in:     danger_of_life(i)

obl(i,behave_cooperatively(i))
exit: NOT cooperate(i) => forb(i,get_info(i))
```

The prima facie norms also have a component "in". However, in contrast to the obligations in the contracts the norms do not arise from actions but arise in certain situations. The prima facie norms do not have a goal because they remain valid as long as the situation in which they arise stays valid.

Because the prima facie norms are by definition general it can easily happen that they conflict in particular situations. For instance, "One should obey ones superior officer" and "One should not kill" will conflict when the officer commands the soldier to kill someone. Some mechanism is needed to determine which of the obligations should be followed in each actual situation. This mechanism determines the actual norms according to

which an agent should behave according to the convention level. In [1] such a mechanism is described in detail and we will not go deeper into this issue here. In the next section we will see that the (remaining) actual norms of the convention level will be compared with the agents private norms and goals to determine its actual behaviour.

4.3 Generation and enforcement of conventions

The interpretation rules do not need enforcement. It seems that one of the advantages of the agent domain above the human domain is that these rules can be fixed. In the human domain the interpretation rules are used by judges to determine the outcome of law cases and are based on experience and personal judgement.

The prima facie norms should be enforced somehow. In the human society they are enforced by a special entity "the state". The state has the power to sanction the violation of norms. If conventions are to be enforced in a multi-agent system it should also contain a central entity that has the power to enforce sanctions on agents that violate the norms. It is still an open question how this should be realized. One possibility is to build a special police agent that has this power in the system. Any agent that wants to coordinate its actions with agents operating within this system must first recognize the power of the police agent. (Similar to pledging allegiance to the state in several nationalization protocols).

In this case the prima facie norms can be modeled in deontic logic using directed obligations, prohibitions and permissions. The counterparty of each obligation would be the police agent, while the obligation is universally quantified over all agents. For instance, "Agents should pay their debts" is modeled as:

$$\forall i O_{ip}(pay - debt)$$

The generation of the conventions can be done in several ways. Most easy is to fix them when the system is started up. This is done in [16] where conventions are modeled explicitly but subsequently are hard-wired into the agent behaviour.
There are two advantages of having the convention level instead of just incorporating the conventions in the communication protocols. The first is that the conventions are now explicit and can be more easily changed.
The second is that there can be many conflicting general conventions. For a particular situation a set of (non-conflicting) actual norms remains. This means that the coordination between agents can change in different circumstances.

5 The Private Level

The notion of a private normative level is necessary to ensure the autonomy of the agents. If agents would only behave according to the higher normative levels, their behaviour would be completely pre-determined by the social norms. Of course an agent with private norms can still always behave within the social norms. In this case its private norms do not conflict with the social norms. This case is different, however, from the case where agents do not have private norms and only look at social norms to direct their behaviour.

We assume that agents have both a pro-active and a reactive behaviour. The pro-active behaviour is determined by the purpose for which the agent is designed. For instance, "collect information about agents on the WWW". This general inherent goal of the agent is translated into a set of conditional preferences for the agent. These preferences are conditional to make them dependent on the environment of the agent. An agent might prefer to collect data from a large bibliography database on another continent at night and might prefer to collect its data from other sources during the day due to transport times and costs.
The conditional preferences are modeled as described in [2, 12]. Conform these definitions we can define preferences as follows:

$Pref_i(\phi|\psi)$ **iff** agent i prefers ϕ to be true in every situation in which ψ is true.

For the private level we did not define a formal description language. Both the contracts as the conventions have to be accessible for several agents. Therefore it is important that they are structured in a uniform way for all agents. The private level, however, is invisible for other agents and the way that the concepts are implemented on this level may vary from one agent to another.

Although it is possible to have agents learn and henceforth adjust their preferences, we assume for the moment that these pro-active preferences are fixed.

Besides the preferences of the agent that arise from the overall purpose of the agent, the agent also has preferences arising from the convention and contract level. These preferences determine the reactive behaviour of the agent. The preferences arising from the contracts and conventions follow from the following axiom schema:

$$\forall i, j Pref_i(\phi|O_{ij}(\phi))$$

I.e. in situations where an agent i has an obligation $O_{ij}(\phi)$ it prefers ϕ to be true. Of course this does not mean that the agent will not violate this obligation. E.g. if an agent is obliged to pay for a flight that it reserved, it will prefer a situation where the flight is payed. However, it might be that the agent does not have enough money yet and does not pay anyway. So, an agent can act against a preference whenever there is a constraint preventing him to do so or if there is another preference with higher priority (we will come back to these priorities shortly) which prevents him from doing so.

The *goals* of an agent are derived from its preferences. However, the goals of an agent are supposed to direct the behaviour of the agent. Therefore any preferences of an agent that are already true are not considered to be goals of the agent. Also preferences that cannot be "achieved" by the agent are not goals because the agent will not attempt to reach unachievable goals. In this case "achieve" is taken in a broad sense. That is, the preference cannot be achieved by the agent acting by it self but also not in cooperation with other agents. In fact, it means that the agent is not capable of devising a plan to reach the preferred situation. Formally the notion of achievability of a situation is defined as follows:

$$Achiev_i(\phi) \equiv \exists \beta : [\beta(i)]\phi \wedge OPP(\beta(i))$$

which means that agent i can achieve ϕ iff there exists a (sequence of) action(s) that causes ϕ to become true and for which i has the OPPortunity to perform it.

The set of goals of an agent can thus be defined as:

The *goals* of an agent are the preferences of an agent which are not true and which are achievable.

This is formally defined as:

$$Goal_i(\phi|\psi) \equiv Pref_i(\phi|\psi) \wedge \neg\phi \wedge Achiev_i(\phi)$$

Note that most preferences that stem from conventions are not goals of the agent because they are not achievable situations for the agent.

In order for the agent to be able to decide which goal it will pursue the goals should be ordered. This is done indirectly by ordering the states according to their (partial) fulfillment of all preferences. Like in [2, 12] we use a utility function on the states for this purpose. The utility function contains some metric to measure how close a state is to the fulfillment of a preference and also a weight of that preference. The goal which is true in the state with the highest utility is chosen by the agent to be the first goal to be pursued. The use of a utility function for the reasoning about preferences indicates that the formal description of these inferences amounts to a non-monotonic logic.
It is important that we do not just order the goals but we consider the goals in light of the combination of all preferences. In this way we move into a direction of "general" preference instead of towards a single important goal.

Of course, the way an agent behaves is determined for a large part by the definition of the utility function. If some conventions indicating cooperative behaviour get a high weight then the agent will tend to behave cooperatively. Also when preferences stemming from contracts of the agent get high weights the agent will react quickly on social obligations.

5.1 Plans, Agenda and Commitment

Once a particular goal has been selected by the agent it has to devise a plan to reach the goal. Usually many plans are possible to reach the goal. Here we do not want to go into the planning process in detail as it is a research area in itself. We just want to mention some considerations that play a role.
First of all agents might have a set of "standard" plans to reach some type of goals. They do not have to start from scratch each time. A second consideration is whether a plan should be "optimal" or should be generated fast. Usually a fast generated "reasonable" plan is preferred to an optimal plan that takes hours to generate. A last consideration is that some plans might be preferred to other plans due to the type of actions they contain and the social norms (e.g. stealing a product vs. buying it).

Once a plan has been devised the tasks of that plan can be put on the agenda of the agent. We consider the agenda of an agent to be the set of intentions of the agent. The tasks on the agenda are intentions because it is not certain whether the agent will actually perform all tasks that are on its agenda at a certain moment.
At the moment the tasks are put on the agenda, the agent also commits itself to the tasks. I.e. it creates an obligation (towards itself or an agent for which it tries to achieve the goal) to perform the tasks. This has been formally described in [8]. It has an advantage over [3] in the fact that the commitment is made explicit in an obligation. The "weight"

of this obligations can be used to order the tasks on the agenda.
If the personal obligation leads to a very high preference then an agent will hardly react to requests from other agents before it finishes the tasks on its agenda. I.e. it might respond to a request but place the actions necessary to fulfill the request at the bottom of its agenda. In this case we say that the agent is very "committed" to its tasks. It only stops if a goal is reached or becomes unachievable.
In the other hand the agent might give its personal obligations a low priority. This leads to agents that easily perform tasks for other agents in between of their own tasks. This way of modeling the agenda is very close to the work of Kinny and Georgeff in [11]. In this work they consider open and closed minded agents. Closed minded agents do not reconsider their goals after the plan has been put on the agenda. Open minded agents check after each step whether the goal is still reachable and whether more important goals appeared.

Although we have given a kind of procedural account of the private norm level of an agent we have not given an explicit architecture of the private level. Due to space limitations we suffice to point to [23, 16] for some global view of the architecture of agents in this framework.

6 Conclusions

In this paper we have given an overview of the concepts that are used to model the social norms that govern the behaviour of autonomous agents. We have shown that many interrelated concepts are needed to capture (part of) the behaviour of autonomous agents. Most of these concepts can be modeled using deontic notions.

The use of deontic logic captures both the autonomy of the agents as well as the (social) dependencies between agents.

Dividing the concepts over three levels makes it possible to structure the different social interactions of an agent. Although the (directed) obligation plays a central role it is accompanied by different concepts on each level and takes a slightly different form on each level as well.
On the convention level the obligations are prima facie norms, on the contract level they are real directed obligations and on the private level they take the form of commitments.

We have given some indications towards the implementation of the norms into a multi-agent system where the agents are based on a type of BDI architecture, where the Desires take the form of preferences and the Intentions are items on the agenda of the agent.
Due to shortage of space and time many questions with regard to an actual implementation remain open. However, no matter how the models will be implemented there are two aspects that are important. First, the norms can be represented explicitly and thus easily be checked and changed. Secondly, there is room for violation of norms and thus for an explicit decision about which norms to adhere to in every situation and how to react when the norms are violated.

In this framework several modal operators have been introduced. Each of these operators are exponents of their own logic. It is still an open question how all these concepts fit into one logic. Maybe each level should have its own logic and some "bridge

rules" should be defined to connect the different levels. This is done in [14].
We are currently working on a first implementation of the agent architecture to check whether our ideas also work in practice.

References

[1] N. Asher and D. Bonevac. Prima Facie Obligation *Studia Logica*, vol.57(1), pages 19-45, 1996.

[2] C. Boutilier. Toward a Logic for Qualitative Decision Theory. In Jon Doyle, Erik Sandewall and Pietro Torasso (eds.), *Principles of Knowledge Representation and Reasoning, proceedings of the fourth international conference*, pages 75-86, 1994, Morgan Kaufmann Publishers, San Francisco, California.

[3] P. Cohen and H. Levesque. Intention is choice with commitment. *Artificial Intelligence*, vol.42, pages 213-261, 1990.

[4] P. Cohen and H. Levesque. Teamwork *Nous*, vol.35, pages 487-512, 1991.

[5] R. Davis and R. Smith. Negotiation as a metaphor for distributed problem solving. *Artificial Intelligence*, vol.20, pages 63-109, 1983.

[6] F. Dignum and R. Kuiper. Combining dynamic deontic logic and temporal logic for the specification of deadlines. HICSS/Logic-Modeling, Hawaii, January 1997.

[7] F. Dignum and H. Weigand. Modeling communication between cooperative systems In J. Iivari, K. Lyytinen and M. Rossi, eds, *Advanced information systems engineering*, pages 140-153, Springer, Berlin, 1995.

[8] F. Dignum, J.-J.Ch. Meyer, R. Wieringa and R. Kuiper. A modal approach to intentions, commitments and obligations: intention plus commitment yields obligation. In M. Brown and J. Carmo (eds.),*Deontic logic, agency and normative systems*, Workshops in Computing, Springer-Verlag, pages 80-97, 1996.

[9] H. Herrestad and C. Krogh Deontic Logic relativised to bearers and counterparties. In J. Bing and O. Torrund, eds, *Anniversary Anthology in Computers and Law*, pages 453-522, Tano A.S., 1995.

[10] N. Jennings. Commitments and Conventions: The foundation of coordination in Multi-Agent systems. *Knowledge Engineering Review*, vol. 8(3), pages 223-250, 1993.

[11] D. Kinny and M. Georgeff. Commitment and Effectiveness of Situated Agents. In *Proceedings International Joint Conference on Artificial Intelligence*, Sydney, Australia, pages 82-88.

[12] J. Lang. Conditional Desires and Utilities - an alternative logical approach to qualitative decision theory. In W. Wahlster, editor, *Proceedings of ECAI-96*, pages 318-327, Budapest, Hungary, 1996, John Wiley & Sons Ltd.

[13] J. Muller. A cooperation model for autonomous agents. In J. P. Müller, M. J. Wooldridge, and N. R. Jennings, editors, *Intelligent Agents III — Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages (ATAL-96)*, Lecture Notes in Artificial Intelligence. Springer-Verlag, Heidelberg, 1996.

[14] P. Noriega and C. Sierra. Towards layered dialogical agents. In J. P. Müller, M. J. Wooldridge, and N. R. Jennings, editors, *Intelligent Agents III — Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages (ATAL-96)*, Lecture Notes in Artificial Intelligence. Springer-Verlag, Heidelberg, 1996.

[15] T. J. Norman, N. R. Jennings, P. Faratin, and E. H. Mamdani. Designing and implementing a multi-agent architecture for business process management. In J. P. Müller, M. J. Wooldridge, and N. R. Jennings, editors, *Intelligent Agents III — Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages (ATAL-96)*, Lecture Notes in Artificial Intelligence. Springer-Verlag, Heidelberg, 1996.

[16] S. Ossowski, A. Garcia-Serrano and J. Cuena. Emergent cooperation of flow control actions through functional cooperation of social agents. In W. Wahlster, editor, *Proceedings of ECAI-96*, pages 539-543, Budapest, Hungary, 1996, John Wiley & Sons Ltd.

[17] A.S. Rao and M.P. Georgeff. Modeling rational agents within a BDI-architecture. In J. Allen, R. Fikes and E. Sandewall, eds, *Proceedings 2d Int. conference on principles of knowledge representation and reasoning*, pages 473-484, San Mateo CA, Morgan Kaufmann, 1991.

[18] W. Ross. The Right and the Good. Oxford University Press. 1930.

[19] L. Royakkers. Representing Legal Rules in Deontic Logic. Ph.D. Thesis, Tilburg University, The Netherlands.

[20] G. Sandu. Reasoning about collective goals. In J. Muller, M. Wooldridge and N. Jennings, eds, *Proceedings ATAL-96*, pages 35-47, Budapest, Hungary, 1996.

[21] J.R. Searle. Speech Acts. Cambridge University Press. 1969.

[22] J.R. Searle and D. Vanderveken. Foundations of illocutionary logic. Cambridge University Press. 1985.

[23] E. Verharen, F. Dignum and H. Weigand. A Language/Action perspective on Cooperative Information Agents. In E. Verharen, N. van der Rijst and J. Dietz, eds, *Proceedings International Workshop on Communication Modeling (LAP-96)*, pages 40-53, Oisterwijk, The Netherlands, 1996.

[24] H. Weigand, E. Verharen and F. Dignum. Interoperable Transactions in Business Models: A Structured Approach. In P. Constantopoulos, J. Mylopoulos and Y. Vassiliou, eds, *Advanced Information Systems Engineering (LNCS 1080)*, pages 193-209, Springer, 1996.