Dynamic Logic for Reasoning about Actions and Agents

J.-J.Ch. Meyer
Utrecht University
Department of Computing Science
Intelligent Systems Group
P.O. Box 80.089
3508 TB Utrecht, The Netherlands

October 17, 1999

Abstract

Dynamic logic is a logic to reason about the dynamics of (natural or artificial) systems in general, ranging from the effects of actions of human agents to the behaviour of artificial agents and software systems. Therefore it is to be expected that in AI it can be fruitfully employed both to represent knowledge about the dynamics of the domain at hand as well as to describe / specify (the dynamic behaviour of) AI systems themselves. A typical example of the former is the description of the effects of actions (of humans, for example) in the commonsense world, while the specification of a particular reasoning system would be of the latter type. In this paper a number of examples are given to illustrate the usefulness (and wide scope!) of dynamic logic for AI.

1 Introduction

Originally, dynamic logic has been proposed in computer science as a logic for reasoning about programs in order to verify their correctness. Later it was realised that dynamic logic could also be fruitfully employed as a logic for reasoning about actions more in general. Recently in AI and computer science the concept of an intelligent agent has become in the limelight very much, with both intelligent robots and software agents (softbots) as intended applications. Since the various attitudes an intelligent agent is supposed to possess can be captured as actions operating on a complex mental state, for the logical description of (the behaviour of) agents one might also resort to dynamic logic (which should then be combined with other modal logics to cater for a description of an agent's mental state).

In this paper some examples are given we have encountered during our research in the past decade illustrating how dynamic logic might constitute a basis of a logic for reasoning about actions and agents. These include a range of applications, from database updates to the description of agent attitudes (belief revision, commitments, obligations) and reasoning about actions in a commonsense environment, where we touch upon the infamous frame problem. The paper contains a rather personal choice of topics and the paper is certainly not meant as a complete overview of dynamic logic and its uses. Furthermore, the topics that are treated are only sketched, without providing all the sordid details, to get a flavour of them, since a full treatment is beyond the scope of the present paper. (For this the reader is referred to other papers.)

2 Dynamic Logic As a Basic Logic of Actions

Dynamic logic is a modal logic especially designed to reason about actions. Historically it dates back to work by Vaughan Pratt [40], Bob Moore [39], and David Harel [19, 18], and it has been used for reasoning about programs, thus providing a formalism for program verification and specification [25, 6]. [1] It is very much akin to Hoare's logic [20] for program correctness, and can in fact be considered as a generalisation of this logic.

In this section we will treat the basic idea behind an elementary form of (propositional) dynamic logic[2], which will serve as a basis for the later logics in this paper.

2.1 Language

For the purpose of this basic treatment we introduce the following logical language $\mathcal{L}_{DL}$. Assume a set $\mathcal{P}$ of propositional atoms, and a set $\mathcal{A}$ of atomic actions. We will use p, q (with possible marks and indexes) to denote elements from $\mathcal{P}$, and a, b (with possible marks and indexes) to denote elements from $\mathcal{A}$.

Definition 2.1 *The logical language $\mathcal{L}_{DL}$ and action language $\mathcal{L}_{ACT}$ are given as the least sets closed under the clauses:*

1. $\mathcal{P} \subseteq \mathcal{L}_{DL}$
2. $\mathcal{A} \subseteq \mathcal{L}_{ACT}$
3. $\varphi, \psi \in \mathcal{L}_{DL}$ *implies* $\neg\varphi, \varphi \wedge \psi, \varphi \vee \psi, \varphi \rightarrow \psi, \varphi \leftrightarrow \psi \in \mathcal{L}_{DL}$

[1] We note that the term "dynamic logic" is somewhat overloaded, since in particular the Amsterdam School in logic employs the term for a dynamic interpretation of classical (non-modal) logic, mostly used in linguistic applications, such as for a formal treatment of discourses [16]. This branch of "dynamic logic" will not be treated in this paper, although we'll touch on it when we'll consider multiple agent updates in section 7.2.1.

[2] Propositional Dynamic Logic or PDL is due to Fischer & Ladner [13]

4. $\varphi \in \mathcal{L}_{DL}, \alpha \in \mathcal{L}_{ACT}$ *implies* $[\alpha]\varphi, \langle\alpha\rangle\varphi \in \mathcal{L}_{DL}$

5. $\varphi \in \mathcal{L}_{DL}$ *implies* $\varphi? \in \mathcal{L}_{ACT}$

6. $\alpha, \beta \in \mathcal{L}_{ACT}$ *implies* $\alpha;\beta, \alpha+\beta, \alpha^* \in \mathcal{L}_{ACT}$

Here we see that the logical language $\mathcal{L}_{DL}$ is an extension of propositional logic with modal operators of the form $[\alpha]$ (and duals $\langle\alpha\rangle$) where α is an element from the action language $\mathcal{L}_{ACT}$. This action language is here taken to be a very basic programming language, viz. that of the regular expressions over 'action alphabet' A. The ';' operator denotes sequential composition ('followed by'), '+' means (nondeterministic) choice, and '*' denotes arbitrary finite repetition. (This choice of operators is 'classic' for dynamic logic (cf. [18, 25, 50]); in this paper also different operators such as '&' (parallel composition) and '-' (action negation) are employed, but in this section we'll restrict ourselves to the familiar ones.) Finally the *action* expression $\varphi?$ stands for a test whether the *logical* expression φ holds in the current state. This is typically used in an expression involving a choice operator to guide control of this choice, as in e.g. the expression $p?;a+\neg p?;b$, where $p \in \mathcal{P}$ and $a, b \in \mathcal{A}$.

2.2 Models

The language $\mathcal{L}_{DL}$ is given an interpretation on the basis of Kripke-models, as is usual for a modal language. We will use tt and ff for the truth values.

Formally a Kripke model for language $\mathcal{L}_{DL}$ is a structure of the following form:

Definition 2.2 *A Kripke model for* $\mathcal{L}_{DL}$ *is a structure* $\mathcal{M}$ *of the form* $\langle S, \pi, r\rangle$, *where*

- S *is a non-empty set (the set of* states*);*
- $\pi : S \to (\mathcal{P} \to \{tt, ff\})$ *is a truth assignment function to the atoms per state;*
- $r : \mathcal{L}_{ACT} \to 2^{S \times S}$ *are state transition relations per action, satisfying the following properties:*
  1. $r(\varphi?) = \{< s, s > \mid \mathcal{M}, s \models \varphi\}$, *where* $\mathcal{M}, s \models \varphi$ *is defined below;*
  2. $r(\alpha;\beta) = r(\alpha) \circ r(\beta)$, *where* $\circ$ *stands for the relational composition;*
  3. $r(\alpha+\beta) = r(\alpha) \cup r(\beta)$;
  4. $r(\alpha^*) = r(\alpha)^\star$, *where* * *stands for the reflexive, transitive closure operator on relations*

Truth of a formula $\varphi \in \mathcal{L}_{DL}$ in a state $s \in S$ in a model $\mathcal{M} = \langle S, \pi, r\rangle$ (written $(\mathcal{M}, s) \models \varphi$), is defined by the following.

Definition 2.3 *Let $\mathcal{M} = \langle S, \pi, r\rangle$ be a given model and $s \in S$. Then:*

- $\mathcal{M}, s \models p$ *iff* $\pi(s)(p) = tt$, *for* $p \in \mathcal{P}$;
- $\mathcal{M}, s \models \neg\varphi$ *iff not* $\mathcal{M}, s \models \varphi$;
- $\mathcal{M}, s \models \varphi \wedge \psi$ *iff* $\mathcal{M}, s \models \varphi$ *and* $\mathcal{M}, s \models \psi$;
- $\mathcal{M}, s \models \varphi \vee \psi$ *iff* $\mathcal{M}, s \models \varphi$ *or* $\mathcal{M}, s \models \psi$;
- $\mathcal{M}, s \models \varphi \rightarrow \psi$ *iff* $\mathcal{M}, s \models \varphi$ *implies* $\mathcal{M}, s \models \psi$;
- $\mathcal{M}, s \models \varphi \leftrightarrow \psi$ *iff* $\mathcal{M}, s \models \varphi$ *bi-implies* $\mathcal{M}, s \models \psi$;
- $\mathcal{M}, s \models [\alpha]\varphi$ *iff* $\mathcal{M}, s' \models \varphi$ *for all* s' *with* $r(\alpha)(s, s')$;
- $\mathcal{M}, s \models \langle\alpha\rangle\varphi$ *iff* $\mathcal{M}, s' \models \varphi$ *for some* s' *with* $r(\alpha)(s, s')$;

A formula φ is *valid in a model* $\mathcal{M} = \langle S, \pi, r\rangle$, denoted $\mathcal{M} \models \varphi$, if $\mathcal{M}, s \models \varphi$ for every $s \in S$. A formula φ is *valid* with respect to a set MOD of models, denoted $MOD \models \varphi$, if $\mathcal{M} \models \varphi$ for every model $\mathcal{M} \in MOD$. If MOD is the set of *all* Kripke models of the above form, we generally write $\models \varphi$ instead of $MOD \models \varphi$.

2.3 The Logic PDL

The simple propositional dynamic logic introduced above can be finitely axiomatized by the following system:

- any axiomatisation of propositional logic
- $[\alpha](\varphi \rightarrow \psi) \rightarrow ([\alpha]\varphi \rightarrow [\alpha]\psi)$
- $[\varphi?]\psi \leftrightarrow (\varphi \rightarrow \psi)$
- $[\alpha;\beta]\varphi \leftrightarrow [\alpha][\beta]\varphi$
- $[\alpha + \beta]\varphi \leftrightarrow [\alpha]\varphi \wedge [\beta]\varphi$
- $[\alpha^*]\varphi \rightarrow \varphi$
- $[\alpha^*]\varphi \rightarrow [\alpha][\alpha^*]\varphi$
- $[\alpha^*](\varphi \rightarrow [\alpha]\varphi) \rightarrow (\varphi \rightarrow [\alpha^*]\varphi)$
- $[\alpha]\varphi \leftrightarrow \neg\langle\alpha\rangle\neg\varphi$

and rules modus ponens (MP) and

- $\frac{\varphi}{[\alpha]\varphi}$

Theorem 2.4 *The above axiomatic system is sound and complete with respect to the class of Kripke models of Definition 2.2.*

We note that taking different choices for the basic operators in the action language, and especially further extensions to these, has a large influence on the axiomatisation. This is the reason that sometimes a quite different axiomatic system is obtained. However, as the above system is rather standard in the literature (e.g. [50]), we have presented it here as a basis.

3 Dynamic Update Logic

Since dynamic logic is suited *par excellence* for reasoning about the dynamics of systems, it may also be used for the description of performing updates on a database (or information system more in general). In [48] this idea is elaborated. Here the logic Propositional Dynamic Database Logic (PDDL) is proposed for rather simple update programs. These are in fact of the form of regular programs as in general propositional dynamic logic, where the atomic actions are now instantiated to updates of the form Ip and Dp for atoms p, denoting insertion and deletion of this atomic fact, respectively. Actually, there are two versions of these updates, *passive* ones and *active* ones, where the latter are parametrized by a logic program H, serving as a background theory, and which are denoted as $T^H p$ and $D^H p$. Here a logic program is a finite set of program clauses of the form $p \leftarrow p_1, \ldots, p_n$, where all p_i and p are atoms. Given a logic program H, we define the set $Der(H)$ of derived atoms in H as the set of all atoms occurring as a head of some clause in H, and the set $Base(H)$ of base atoms in H as $\mathcal{P} \backslash Der(H)$.

The semantics of this logic is as in the general case, only now the specific atomic actions should be catered for. To this end we need some additional definitions.

Definition 3.1

- *Let $\mathcal{M} = \langle S, \pi, r \rangle$ be a Kripke model. The function prop : $S \rightarrow 2^{\mathcal{P}}$ is given by $prop(s) = \{p \in \mathcal{P} \mid \pi(s)(p) = tt\}$.*
- *A Kripke model $\mathcal{M} = \langle S, \pi, r \rangle$ is* full *iff for all $P \subseteq \mathcal{P}$ there exists a state $s \in S$ such that $prop(s) = P$.*
- *Let H be a logic program. The function $min^H : 2^{\mathcal{P}} \rightarrow 2^{\mathcal{P}}$ is given by: $min^H(P)$ is the minimal set P' (w.r.t. set inclusion) such that $(P \cap Base(H)) \subseteq P'$ and such that if $p \leftarrow p_1, \ldots, p_n \in H$ and $p_1, \ldots, p_n \in P'$ then also $p \in P'$.*

Now we can state the semantics of (passive and active) insertion and deletion:

Definition 3.2

- $r(Ip) = \{< s, s' > \mid prop(s') = prop(s) \cup \{p\}\}$

- $r(Dp) = \{< s, s' > \mid prop(s') = prop(s) \backslash \{p\}\}$
- $r(I^H p) = \{< s, s' > \mid prop(s') = min^H(prop(s) \cup \{p\})\}$
- $r(I^H p) = \{< s, s' > \mid prop(s') = min^H(prop(s) \backslash \{p\})\}$

The axiomatisation of this logic includes the following axioms (besides the familiar ones from PDL):

- $[Ip]p$
- $[Dp]\neg p$
- $\langle Ip \rangle \mathtt{tt}$
- $\langle Dp \rangle \mathtt{tt}$

One also have to deal with non-changes by means of the so-called *frame axioms*:

- $q \rightarrow [Ip]q$, for $q \neq p$
- $\neg q \rightarrow [Ip]\neg q$, for $q \neq p$
- $q \rightarrow [Dp]q$, for $q \neq p$
- $\neg q \rightarrow [Dp]\neg q$, for $q \neq p$

As to active updates we have analogous axioms:

- $[I^H p]p$
- $[D^H p]\neg p$
- $\langle I^H p \rangle \mathtt{tt}$
- $\langle D^H p \rangle \mathtt{tt}$
- $q \rightarrow [Ip]q$, for $q \in Base(H) \backslash \{p\}$
- $\neg q \rightarrow [Ip]\neg q$, for $q \in Base(H) \backslash \{p\}$
- $q \rightarrow [Dp]q$, for $q \in Base(H) \backslash \{p\}$
- $\neg q \rightarrow [Dp]\neg q$, for $q \in Base(H) \backslash \{p\}$

However, to deal with derived atoms we need some more definitions:

Definition 3.3 *Let H be a logic program.*

- *Let $q \in Der(H)$ such that*

$$q \leftarrow p_{11}, \ldots p_{1n_1}$$

$$\ldots$$

$$q \leftarrow p_{m1}, \ldots p_{mn_m}$$

are all program clauses in H with q as head. Then the completion formula *of q in H, denoted H_q, is defined as:*

$$H_q = (p_{11} \wedge \ldots \wedge p_{1n_1}) \vee \ldots \vee (p_{m1} \wedge \ldots \wedge p_{mn_m})$$

- *The binary relation $dep_H \subseteq \mathcal{P} \times \mathcal{P}$ is defined as: $(p,q) \in dep_H$ iff there is a program clause $p \leftarrow q_1, \ldots, q_n \in H$ such that $q = q_i$ for some i. The relation dep_H^+ is the transitive closure of dep_H.*

- *An atom p is said to be* recursive *in H if there is an atom q such that $(p,q) \in dep_H^+$ and $(q,q) \in dep_H^+$. The set of recursive atoms in H is denoted $Rec(H)$. H is called recursive if $Rec(H) \neq \emptyset$.*

- *A* mutual dependency group *of H is non-empty subset P of $rec(H)$ such that $\forall p, q \in P : (p,q) \in dep_H^+$.*

Now we have the following axioms pertaining to derived atoms:

- $[I^H p](q \leftrightarrow H_q)$, for $q \in Der(H)$
- $[D^H p](q \leftrightarrow H_q)$, for $q \in Der(H)$
- $[I^H p](\vee_{q \in P} q \rightarrow \vee_{q \in P} H_q[\mathtt{ff}/P])$
  for P a mutual dependency group of H
- $[D^H p](\vee_{q \in P} q \rightarrow \vee_{q \in P} H_q[\mathtt{ff}/P])$
  for P a mutual dependency group of H

(Here $\varphi[\mathtt{ff}/P]$ stands for the formula φ in which all occurrences of atoms $p \in P$ are replaced by `ff`.)

The last two axioms are minimization axioms: they state that if recursive atoms are true, then there must be an "external reason" for this; the atoms in a mutual dependency group cannot be true only because other atoms in this group are true. If a recursive atom in such a group is true, then the completion formula for some atom in the group must be true, even if the recursive atoms in the completion formula are replaced by `ff`.

In [48] it is shown that this axiomatization of PDDL is complete with respect to so-called *full* structures. (In that paper also a complete axiomatization is given for *all* structures, but we will not elaborate on this here.)

Next in [49] this idea is generalised to the first-order case, which of course in the context of databases is almost a 'must'. In fact in this paper two update logics are proposed, both based on (first-order) dynamic logic.

The first one is Relational Algebra Update Logic (RAUL), which can be viewed as an extension of relational algebra with assignment. The second logic is Dynamic Database Logic (DDL) in which the atomic updates are updates of the extension of predicates and assignments, which are updates to the values of attributes construed as updatable functions. DDL can be viewed as a true extension of PDDL to the first-order case. We will not go into this here further, but refer to [49] for this, and restrict ourselves here to a brief sketch of RAUL.

In RAUL we consider as actions typically assignments of the form $p := e$, where p is a predicate symbol, and e is a relational algebra expression, which has a relation (i.e., a set of tuples) as its meaning. Relational algebra expressions are of the form $e_1 \cup e_2, e_1 \backslash e_2, e_1 \times e_2, e[k_1, \ldots, k_n], e$ `where` φ, denoting union, difference, product, projection and selection (here φ is a test with which one can test whether attributes are equal to a term or to each other), respectively. We omit here the formal semantics of these expressions, and go straight to the logic. Typical axioms in RAUL include for example:

- $qT \rightarrow [p := q]pT$
- $\neg qT \rightarrow [p := q]\neg pT$
- $[p := e_1 \cup e_2]pT \leftrightarrow ([p := e_1]pT \vee [p := e_2]pT)$
- $[p := e_1 \backslash e_2]pT \leftrightarrow ([p := e_1]pT \wedge \neg[p := e_2]pT)$
- $[p := e_1 \times e_2]pT_1T_2 \leftrightarrow ([p_1 := e_1]p_1T_1 \wedge [p_2 := e_2]p_2T_2)$
- $\langle p := e\rangle \mathtt{tt}$

Again one also have to deal with non-changes by means of frame axioms:

- $qT \rightarrow [p := e]qT$, for $q \neq p$
- $fT = t \rightarrow [p := e]fT = t$

expressing that, by assigning to an updatable predicate, other updatable predicates and updatable functions do not change.

4 Dynamic Semantics of Reasoning Systems

In the previous section we have seen how dynamic logic can be applied to describe the dynamics of updating a database or an information system more in general. In the case of active updates this also involved some reasoning already due to the logic program stored in the information system that served as a

background theory, in particular pertaining to derived atoms. Of course one may turn to a more advanced system of reasoning. Also such more advanced reasoning systems themselves can be analyzed by means of dynamic logic. The basic idea behind this is that reasoning systems like any other dynamic system display a certain behaviour that can also be thought of being brought about by reasoning steps. These may be 'classical steps' like applying deduction, but – and in AI this is even more important – also reasoning steps in a 'non-standard' logic or reasoning system. One may, for instance, think of default reasoning, but there are many other examples as well.

4.1 Descriptive Dynamic Logic

For example, Sierra *et al.* [46] have applied dynamic logic for analysing / describing reasoning by means of a reflective (multi-language) architecture, where typical reasoning steps include 'reflection' steps, such as moving from an object level to a meta-level where one e.g. may reason about one's knowledge and ignorance on the assertions on the object level. More in general in multi-language architectures one employs so-called *bridge rules* to infer / transfer / translate information between two different *units* (that is to say, languages or, if one thinks more in terms of an implementation of such an architecture, modules). Of course also inferences within a unit may take place.

In their set-up called Descriptive Dynamic Logic they tailor (propositional) dynamic logic to suit their purpose by taking as atomic formulas so-called "quoted" formulas of the form $k : \lceil\varphi\rceil$, where k is a unit identifier denoting a unit u_k of the system, and φ is a formula in the language of that unit. 'Subatomic' programs are inferences (deduction steps) by means of (intra-unit or inter-unit) inference rules, written as $\lceil\Gamma \vdash_{kl} \varphi\rceil$, where k and l refer to the units u_k and u_l in the system, respectively (if $k = l$, we have an intra-unit inference), and Γ is a set of formulas in the language of u_k and φ is a formula in the language of u_l.

Atomic programs in the sense of propositional dynamic logic are now taken to be non-deterministic choices of subatomic actions.[3] For instance, we have atomic actions like $\vdash_{kl} = \bigcup\{\alpha \mid \alpha \text{ is of the form } \lceil\Gamma \vdash_{kl} \varphi\rceil\}$, where $\bigcup$ stands for the nondeterministic choice of the subatomic actions that are given to it as argument.

In this language one may express the case where a unit u_k is endowed with the modus ponens inference rule by the (valid) formula:

$$MP : \langle\beta\rangle(k : \lceil\varphi\rceil \wedge k : \lceil\varphi \rightarrow \psi\rceil) \rightarrow \langle\beta; \vdash_{kk}^{mp}\rangle k : \lceil\psi\rceil,$$

where the modus ponens program $\vdash_{kk}^{mp} = \bigcup_{\gamma,\delta} \lceil\{\gamma, \gamma \rightarrow \delta\} \vdash_{kk} \delta\rceil$.

Other typical axioms in this logic include:

[3] We employ here a somewhat different terminology than in [46] to stay in line with the rest of our paper.

- $\langle\lceil\Gamma \vdash_{ij} \psi\rceil\rangle\mathtt{tt} \leftrightarrow \bigwedge_{\varphi\in\Gamma} i : \lceil\varphi\rceil$
- $[\lceil\Gamma \vdash_{ij} \psi\rceil]j : \lceil\psi\rceil$
- $\langle\lceil\Gamma \vdash_{ij} \psi\rceil\rangle p \rightarrow p$, for atomic proposition $p \neq j : \lceil\psi\rceil$

The first one expresses that an application of a deduction step (viewed as a program) $\lceil\Gamma \vdash_{ij} \psi\rceil$ succeeds if and only if the formulas in the antecedent Γ are all established within unit u_i. The second one says that an application of the deduction step $\lceil\Gamma \vdash_{ij} \psi\rceil$ (when it succeeds) results in a state where the conclusion ψ is established in unit u_j. The third is again a kind of frame axiom: after application of a step $\lceil\Gamma \vdash_{ij} \psi\rceil$ every atom p different from $j : \lceil\psi\rceil$ that is true was already true before application (informally, of course, since the step has no bearing on such an atom).

In [46] a complete axiomatisation of the logic is given. Furthermore it is also indicated how this logic might be used to describe a reflective architecture for non-standard reasoning methods like default reasoning.

5 Dynamic Deontic Logic

In this section we consider another application of dynamic logic, viz. reasoning about the deontics of actions. Deontic logic itself is an old branch of philosophical (modal) logic dating back to Mally [32], but which has become a serious subject of study since the seminal work of Von Wright [54, 55]. Deontic logic is the logic of obligation, prohibition, and permission and can, in principle, be applied both to obligated / forbidden / permitted states of affairs and obligated / forbidden / permitted actions. In the literature these two forms of deontic logic have been referred to as *ought-to-be* and *ought-to-do* logic, respectively, and it is mostly not made explicit which form is treated / meant. In [34] an explicit ought-to-do logic has been proposed, based on dynamic logic. The main idea here was inspired by work of Anderson [1], who tried to reduce deontic logic to alethic modal logic. The crux of his idea was that something (φ) is forbidden iff violation of the constraint φ has some undesirable effect. In [34] this idea is applied explicitly to actions: an action if forbidden iff performing it leads to an undesirable effect. This is easily formalised in dynamic logic.

5.1 The Logic PD_eL

In order to reason about the deontics of actions we take the following language, where we assume a special propositional constant $\mathbf{V} \in \mathcal{P}$, denoting a state of 'violation' of a deontic constraint.

Definition 5.1 *The logical language $\mathcal{L}_{DeL}$ and action language $\mathcal{L}_{ACT0}$ are given as the least sets closed under the clauses:*

1. $\mathcal{P} \subseteq \mathcal{L}_{DeL}$

2. $\mathcal{A} \subseteq \mathcal{L}_{ACT0}$

3. $\varphi, \psi \in \mathcal{L}_{DeL}$ *implies* $\neg\varphi, \varphi \wedge \psi, \varphi \vee \psi, \varphi \rightarrow \psi, \varphi \leftrightarrow \psi \in \mathcal{L}_{DeL}$

4. $\varphi \in \mathcal{L}_{DeL}, \alpha \in \mathcal{L}_{ACT0}$ *implies* $[\alpha]\varphi, \langle\alpha\rangle\varphi \in \mathcal{L}_{DeL}$

5. $\alpha, \beta \in \mathcal{L}_{ACT0}$ *implies* $\alpha;\beta, \alpha + \beta, \alpha\&\beta, \overline{\alpha} \in \mathcal{L}_{ACT0}$

Thus in the language we use for deontic purposes we employ a slightly different set of action operators, viz. sequential composition, alternative composition (choice), parallel composition and action negation(!). The latter is a little non-standard in dynamic logic. It expresses that the action is *not* performed. This construct will enable us to express the obligation of an action below.

Models for the language $\mathcal{L}_{DeL}$ are like those for $\mathcal{L}_{DL}$, but of course, one has now to cater for the operators & and -. Since we like the (interpretations of) the operators to satisfy the rules of a Boolean algebra, this is (surprisingly) rather involved (due to the presence of the sequential composition ';'), so that we do not give its full definition here. Here it suffices that & and - are very much resembling an intersection and complement operator on the accessibility relations (like + is resembling an union operator) (cf. [34, 9]).[4]

We can now define the deontic operators as abbreviations as follows:

Definition 5.2

- $\mathbf{F}\alpha \equiv [\alpha]\mathbf{V}$, *i.e., the action* α *is forbidden iff performing* α *leads to a state of violation;*
- $\mathbf{P}\alpha \equiv \neg\mathbf{F}\alpha$, *i.e., the action* α *is permitted iff it is not forbidden;*
- $\mathbf{Obl}\alpha \equiv [\overline{\alpha}]\mathbf{V}$, *i.e., an action* α *is obligated iff not-doing it leads to a violation.*

Although the basic ideas behind these definitions are very simple indeed, it leads already to some interesting deontic properties:

Proposition 5.3

1. $(\mathbf{Obl}\alpha \vee \mathbf{Obl}\beta) \rightarrow \mathbf{Obl}(\alpha + \beta)$
2. $\mathbf{P}(\alpha + \beta) \leftrightarrow (\mathbf{P}\alpha \vee \mathbf{P}\beta)$
3. $\mathbf{F}(\alpha + \beta) \leftrightarrow (\mathbf{F}\alpha \wedge \mathbf{F}\beta)$
4. $\mathbf{Obl}(\alpha\&\beta) \leftrightarrow (\mathbf{Obl}\alpha \wedge \mathbf{Obl}\beta)$
5. $\mathbf{P}(\alpha\&\beta) \rightarrow (\mathbf{P}\alpha \wedge \mathbf{P}\beta)$
6. $(\mathbf{F}\alpha \vee \mathbf{F}\beta) \rightarrow \mathbf{F}(\alpha\&\beta)$

[4] With respect to the negation of sequential composition we have that in the semantics it holds that $\overline{\alpha;\beta} = \overline{\alpha} + (\alpha;\overline{\beta})$.

7. $\mathbf{Obl}(\alpha;\beta) \leftrightarrow (\mathbf{Obl}\alpha \wedge [\alpha]\mathbf{Obl}\beta)$

8. $\mathbf{P}(\alpha;\beta) \leftrightarrow \langle\alpha\rangle\mathbf{P}\beta$

9. $\mathbf{F}(\alpha;\beta) \leftrightarrow [\alpha]\mathbf{F}\beta$

Although these properties are rather intuitive, one might consider alternative approaches with (slightly) different ones. For example, in legal reasoning it is not always the case that everything that is not forbidden is automatically permitted, so that one might also consider a class of actions that is neither (explicitly) forbidden nor (explicitly) permitted (cf. e.g. [47]). Sometimes also a property like $(\mathbf{Obl}\alpha \vee \mathbf{Obl}\beta) \rightarrow \mathbf{Obl}(\alpha + \beta)$ and, hence, $\mathbf{Obl}\alpha \rightarrow \mathbf{Obl}(\alpha + \beta)$ is considered undesirable. This property is called Ross's paradox, and it is counterintuitive if one reads the choice operator in such a way that the agent is free to choose its alternatives him/herself: "if one ought to mail the letter, then one ought to mail the letter or burn it" sounds counterintuitive in this reading. Deontic logic traditionally has a history of such paradoxes.[5] Another paradox was discovered by Van der Meyden [33] concerning permission in our approach, where only the 'end' situation is decisive as to whether the action is permitted: $\mathbf{P}(\alpha;\beta) \leftrightarrow \langle\alpha\rangle\mathbf{P}\beta$ is equivalent with $\mathbf{P}(\alpha;\beta) \leftrightarrow \langle\alpha\rangle(\langle\beta\rangle\neg\mathbf{V})$, which has as a consequence that doing α resulting in a violation state while then doing β resolving the violation is permitted according to our definitions. So this is a kind of " the end justifies the means" approach, and while this might be acceptable for some applications, it is obviously not the case for all uses / readings of permission, so that one should then use a stronger notion. In fact several of these alternatives have been proposed. We ourselves have also considered alternatives to deal with these so-called paradoxes (e.g. [9]), but most of them can be viewed harmless once one realizes what exactly the meanings of these are, and one is then able to 'fine-tune' these notions to one's needs.

Deontic logic can be employed for the representation of knowledge in domains where obligations, permissions and prohibitions occur directly, such as the legal reasoning domain [41], but its application goes beyond that. It has also been proved very useful in the description of systems where it is important to distinguish ideal from actual behaviour. Ideal behaviour is then specified by using deontic operators which can then be well distinguished from actual behaviour. So here we see that there are more or less implicit (ideal) constraints that can be (actually) violated by the system. Examples of such systems include fault-tolerant software systems [5], but also information systems such as databases where there may be so-called soft constraints that every system state ought to / should satisfy, but again might be violated [38]. It has recently been recognized that being able to represent such a violation (by e.g. the use of deontic operators) is very helpful to specify for instance a way to recover from such

[5] To be fair, other modal logics have some of this trouble as well; for instance epistemic logic suffers from the so-called logical omniscience problem(s), which is very much akin conceptually and technically to the deontic paradoxes (cf. [36]).

a violation. Soft constraints typically occur in systems where not all actors / agents in the system are under complete control and have a certain *autonomy* so that they can choose themselves to violate these constraints. A commonsense example would be an automated library system where nevertheless the people borrowing books may choose to return their books too late according to the library's regulations.

6 Typical Effects of Commonsense Actions and the Frame Problem

In AI there has been considerable effort to give an adequate treatment of reasoning about actions in a commonsense context. It has appeared to be a very hard problem to describe the effects and particularly the *non*-effects of actions. This problem has become known as the *frame problem*. Related problems include the specification of the necessary preconditions for an action to be performed successfully (or intendedly), also known as the *qualification problem*, and the precise determination of the derived effects of an action, the *ramification problem* (cf. [43]). Although there has been a lot of research on this problem and definitely some progress have been made, I believe that in general the frame problem and its related problems remain highly problematic. The problem is the space of possible (non-) effects and preconditions / qualifications and ramifications is potentially infinite, or at least astronomical in practical cases, so that it is virtually impossible to give such a precise specification in those cases, and neither do we really want to. What is searched for is a 'convenient' method to deal with this problem, which triggered a lot of research in nonmotonic reasoning (such as default reasoning) with the idea in mind that 'by default', unless explicitly specified or deduced otherwise, aspects / features of the world remain the same. (This is sometimes called *'inertia'*, but one has to bear in mind that this has nothing to do with physics, but rather with the reasoner's mental laziness to reckon with all conceivable changes/preconditions/ramifications of the world due to the performance of some action! But this laziness on the part of the reasoner is not to be blamed on him, since s/he has to act in the world in a real-time fashion...)

6.1 Preferential Action Semantics

In [35] an approach to the specification of actions dealing with some of the issues mentioned above is proposed which is based on the use of dynamic logic (inspired by work reported in [31] using the different but related framework of Dijkstra's weakest precondition calculus as well as the Features and Fluents approach as taken in [42]). Essentially this approach consists of specifying per action which aspects of the world, in this area often called *features*, are definitely *set* to some value, which are *framed*, that is subject to *inertia*, and

which fluents are considered to be truly *variable*, expressing that their values may be nondeterministically change during the execution / performance of the action at hand. In the literature on action changes and the frame problem it has proved to be also useful to sometimes *'release'* features temporarily from being framed (i.e. being subject to inertia) so that they become temporarily variable (cf. e.g. [42, 24]).

The action language that we consider here is the following variant of the basic one. Besides a set $\mathcal{A}$ of atomic actions we assume a set $\mathcal{F}$ of *feature* variables.

Definition 6.1 *The action language $\mathcal{L}_{PA}$ is given as the least set closed under the clauses:*

1. $\mathcal{A} \subseteq \mathcal{L}_{PA}$
2. $\alpha, \beta \in \mathcal{L}_{PA}$ *implies* $\alpha \oplus \beta, \alpha + \beta, \alpha \| \beta,$ `if` b `then` α `else` $\beta \in \mathcal{L}_{PA}$, *where b is a boolean test on feature variables*
3. $\alpha \in \mathcal{L}_{PA}$ *implies* $\alpha_{\#} \in \mathcal{L}_{PA}$

Thus here we have atomic actions, and compositions of these by means of the operators $\oplus$ (restricted choice), $+$ (liberal choice), $\|$ (parallel composition) and a conditional composition operator. Moreover we include compositions by means of the operators $\#$ (preferred behaviour). Note the lack of a sequential composition in this language.[6]

To render semantics to these action expressions, we need to fix some things first. First of all, we endow an atomic action $a \in \mathcal{A}$ with a 'signature' (set_a, $frame_a, release_a$), where $set_a, frame_a, release_a \subseteq \mathcal{F}$, and $release_a \subseteq frame_a$, describing the status of the feature variables w.r.t. this action: set_a is the set of feature variables that are *set* by a, $frame_a$ is the set of variables that are *framed* while performing a, of which the variables in $release_a$ are *released* in the sense we mentioned above. For convenience we also use the abbreviations $inert_a = frame_a \backslash release_a$ for the feature variables that are subject to inertia while performing action a and $var_a = \mathcal{F} \backslash (set_a \cup frame_a)$ for the feature variables that are 'truly variable', i.e. are known to be subject to change of their values in a nondeterministic way.

In this context we define a state s as a function: $\mathcal{F} \to \mathcal{V}$, where $\mathcal{V}$ is a set of feature values (in many examples this set may be taken to be the booleans). The set of states is denoted Σ. The state $s\{d/x\}$, with $s \in \Sigma, d \in \mathcal{V}, x \in \mathcal{F}$, is defined as the state s' satisfying $s'(x) = d$, and $s'(y) = s(y)$ for $y \neq x$. For $X \subseteq \mathcal{F}$ and $s \in \Sigma$, we define $vary_X(s) = \{s' \mid s' = s\{v_1/x_1, v_2/x_2, \ldots, v_n/x_n\}$

[6] This operator may be added, but as it is not clear what it means to have the preferred behaviour of a sequentially composed action, one has to take care that the # operator cannot be applied to such an action. Thus a layered language is needed which we will not define here. Here we only note that in the logical language a property of a sequentially composed scenario $\alpha; \beta$, like $[\alpha; \beta]\varphi$, can be expressed by $[\alpha][\beta]\varphi$.

for any subsets $\{v_1, v_2, \ldots, v_n\} \subseteq \mathcal{V}, \{x_1, x_2, \ldots, x_n\} \subseteq X\}$. Furthermore, for $S \subseteq \Sigma$, we put $vary_X(S) = \bigcup_{s \in S} vary_X(s)$. Note that $vary$ has the property that $vary_X(vary_Y(S)) = vary_{X \cup Y}(S)$.

Now we are ready for the formal semantics of our actions expressions:

For atomic actions $a \in \mathcal{A}$ we stipulate an accessibility relation $r(a) \subseteq \Sigma \times \Sigma$ that yields the behaviour of a with respect to the variables in set_a. Thus, $r(a)(s, s')$ captures the setting of the variables in set_a by means of a.

To cater for the other variables, we include in the semantics of an action α (given an input state) a number of items, viz.

1. the set of all states that *possibly* may result as a consequence of performing the action α, when considering all the variables apart from the *set* ones in set_α truly variable,

2. the set of all states that are preferred / expected as a consequence of performing the action α, taking inertia of the *inert* variables in $inert_\alpha$ into account, and

3. a triple $(set_\alpha, frame_\alpha, release_\alpha)$ that records the exact resulting status of the variables during performance of α (for future reference, see below).

For atomic actions this is given by means of the semantic function $[\![\cdot]\!]$: $[\![a]\!](s) =$

$$\{(\bigcup_{s' : r(a)(s,s')} vary_{var_a \cup frame_a}(s'), \bigcup_{s' : r(a)(s,s')} vary_{var_a \cup release_a}(s'), (set_a, frame_a, release_a))\}$$

For compound actions α we also need to establish the status of the variables. We do this again by sets set_α, $frame_\alpha$ and $release_\alpha$, and use abbreviations $inert_\alpha = frame_\alpha \backslash release_\alpha$ and $var_\alpha = \mathcal{F} \backslash (set_\alpha \cup frame_\alpha)$.

The semantics of the conditional composition operator is as usual. With respect to the other operators we define:

$$[\![\alpha \triangle \beta]\!](s) = [\![\alpha]\!](s) \triangle [\![\beta]\!](s)$$

for $\triangle = \oplus, +, \|$ and

$$[\![\alpha_\#]\!](s) = \#([\![\alpha]\!](s))$$

where $Z_1 \triangle Z_2 = \bigcup_{z_1 \in Z_1, z_2 \in Z_2} z_1 \triangle z_2$ for $\triangle = \oplus, +, \|$ and $\#(Z) = \bigcup_{z \in Z} \#(z)$ and

- regarding the operator $\oplus$ we define:

$$(S_1, S_1', (set_1, frame_1, release_1)) \oplus (S_2, S_2', (set_2, frame_2, release_2)) = \{(S_1, S_1', (set_1, frame_1, release_1)), (S_2, S_2', (set_2, frame_2, release_2))\}$$

- regarding the operator $+$ we have:

$$(S_1, S_1', (set_1, frame_1, release_1)) + (S_2, S_2', (set_2, frame_2, release_2)) = \\ \{(S_3, S_3', (set_1, frame_3, release_3)), (S_4, S_4', (set_2, frame_4, release_4))\}$$

where $S_3 = vary_{(set_2 \cup frame_2) \setminus set_1}(S_1)$,
$S_3' = vary_{(set_2 \cup release_2) \setminus set_1}(S_1')$,
$S_4 = vary_{(set_1 \cup frame_1) \setminus set_2}(S_2)$,
$S_4' = vary_{(set_1 \cup release_1) \setminus set_2}(S_2')$,
$frame_3 = (frame_1 \cup frame_2 \cup set_2) \setminus set_1$,
$release_3 = (release_1 \cup release_2 \cup set_2) \setminus set_1$,
$frame_4 = (frame_1 \cup frame_2 \cup set_1) \setminus set_2$,
$release_4 = (release_1 \cup release_2 \cup set_1) \setminus set_2$

- regarding the operator $\|$ we have:

$$(S_1, S_1', (set_1, frame_1, release_1)) \parallel (S_2, S_2', (set_2, frame_2, release_2)) = \\ \{(S_3 \cap S_4, S_3' \cap S_4', (set_3, frame_3, release_3))\}$$

where S_3, S_3', S_4, S_4' are as in the previous case, and $set_3 = set_1 \cup set_2$, $frame_3 = (frame_1 \cup frame_2) \setminus (set_1 \cup set_2)$, $release_3 = (release_1 \cup release_2) \setminus (set_1 \cup set_2)$

- and finally regarding the preference operator $\#$ it holds that:

$$\#(S, S', (set, frame, release)) = (S', S', (set \cup inert, release, release))$$

where $inert = frame \backslash release$.

Note that $[(\alpha_\#)_\#] = [\alpha_\#]$. If, for $[\![\alpha]\!](s) = \{(S_1, S_1', \ldots), (S_2, S_2', \ldots), \ldots\}$, we use the notation $[\![\alpha]\!]_1(s) = \bigcup_i S_i$ and $[\![\alpha]\!]_2(s) = \bigcup_i S_i'$, we may define the interpretation of formulas $[\alpha]\varphi$ and $[\alpha_\#]\varphi$ as follows:

- $s \models [\alpha]\varphi$ iff $t \models \varphi$ for all $t \in [\![\alpha]\!]_1(s)$
- $s \models [\alpha_\#]\varphi$ iff $t \models \varphi$ for all $t \in [\![\alpha]\!]_2(s)$

Typical validities in this framework are:

- $\models [\alpha]\varphi \rightarrow [\alpha_\#]\varphi$
- $\models [\alpha]\varphi \rightarrow [\alpha \| \beta]\varphi$
- $\models [(\alpha + \beta)_\#]\varphi \rightarrow [(\alpha \oplus \beta)_\#]\varphi$

the first expressing that the expected states after execution of an action constitute a subset of the possible states after executing that action; the second that concerning all possible successor states those of α are a superset of those

of the parallel execution $\alpha \| \beta$; and the third expressing that the expected states after the execution of the restricted choice between two actions is a subset of those resulting after a liberal choice.

Typical *non*-validities are:

- $\not\models [\alpha_{\#}]\varphi \rightarrow [(\alpha \| \beta)_{\#}]\varphi$
- $\not\models [(\alpha \oplus \beta)_{\#}]\varphi \rightarrow [(\alpha + \beta)_{\#}]\varphi$

the first stating that in general the set of *expected* states when executing an action is *not* a superset of the set of *expected* states when executing that action in parallel with some other action; the second that the expected states after the execution of the restricted choice between two actions is generally a *proper* subset of those resulting after a liberal choice.

Interestingly, in this approach preferred behaviour in terms of applying inertia when possible is explained in terms of 'concurrency': if no concurrent events (or actions as performed by other agents) are known to interfere, the behaviour of the action at hand in which inertia is applied to the *inert* variables is preferred. Thus, for example, the preferred behaviour of a *wait* action, viewed in isolation is the same as that of the *skip* action, the interpretation of which is the identity function, while viewed in a concurrent context we may have to take into account the interference of other events. (This is different from the approach proposed in [31] where the *wait* action is simply identified with the *skip* action.) In [35] we show how typical AI scenarios involving inertia like the infamous Yale Shooting Scenario can be treated adequately in this set-up.

7 Specifying Intelligent Agents

In the philosophical literature the term 'agent' is often used to refer to the (human) subject who knows, reasons or performs acts. Recently in computer science the term '(intelligent) agent' has become popular to refer to intelligent pieces of software (or hardware) that perform 'intelligent' tasks autonomously, that is, in a way that is not directly dependent on the user's actions / requests [56, 23]. Typical examples of software agents (or softbots) are personal assistents to help and guide the internet user through the ever increasing and perhaps already almost impenetrable jungle of the World Wide Web. Of course, another example of these artificial agents are robots that should be able to reason and perform intelligent tasks. In this section we will use a modal logic based on dynamic logic to describe or specify the behaviour of such intelligent agents, thus providing a theoretic framework on which the realisation of these artefacts may be founded.

7.1 Single Agents

First we will discuss the behaviour / attitudes of a single agent in isolation. In the next subsection we will see what issues have to be tackled when 'societies of agents' are studied. We give a simplified treatment of our work reported in [26, 27, 28, 37].

7.1.1 Agent Attitudes

Typical agent attitudes include, besides the ability of acting(!), possession and maintenance of knowledge / belief, and intentional or motivational ones, such as the possession and maintenance of desires, goals and commitments. These attitudes may be described in a logic that consists of dynamic logic, enriched with (modal) operators dealing with such notions as ability, opportunity, knowledge, belief, desires and goals. So we enrich the language with operators $\mathbf{A}, \mathbf{O}, \mathbf{K}, \mathbf{B}, \mathbf{D}, \mathbf{G}$.[7] Moreover, we introduce special actions `revise` and `commit` in the language to deal with revision of knowledge / belief and the performance of commitments (which can be seen as an operator that revises the agent's agenda).

Definition 7.1 *The logical language $\mathcal{L}_{SA}$ and action language $\mathcal{L}_{ACT,SA}$ are the least extensions of the languages $\mathcal{L}_{DL}$ and $\mathcal{L}_{ACT}$, respectively, that are closed under the clauses:*

1. $\varphi \in \mathcal{L}_{SA}$ *implies* $\mathbf{K}\varphi, \mathbf{B}\varphi, \mathbf{D}\varphi, \mathbf{G}\varphi \in \mathcal{L}_{SA}$
2. $\varphi \in \mathcal{L}_{SA},\ \alpha \in \mathcal{L}_{ACT,SA}$ *implies* $\mathbf{A}\alpha, \mathbf{O}\alpha \in \mathcal{L}_{SA}$
3. $\varphi \in \mathcal{L}_{SA}, \alpha \in \mathcal{L}_{ACT,SA}$ *implies* $[\![\alpha]\!]\varphi, \ll \alpha \gg \varphi \in \mathcal{L}_{SA}$
4. $\varphi \in \mathcal{L}_{SA}$ *implies* $\texttt{revise}\varphi \in \mathcal{L}_{ACT,SA}$
5. $\alpha \in \mathcal{L}_{ACT,SA}$ *implies* $\texttt{commit}\alpha \in \mathcal{L}_{ACT,SA}$

To interpret this extended language we need also to extend our models:

Definition 7.2 *A Kripke model for $\mathcal{L}_{SA}$ is a structure $\mathcal{M}$ of the form $\langle S, \pi, r, t, R_{\mathbf{K}}, R_{\mathbf{B}}, R_{\mathbf{D}}, Agenda \rangle$, where*

- *S is a non-empty set (the set of* states*);*

[7] We mention here also related work [22] in which a so-called preference-based action logic is proposed based on dynamic logic in which goals are expressed as well. The main difference between this logic and ours is that Huang *et al.* use a blend of dynamic logic, preference logic (where preferences are taken in the sense of Herbert Simon), (a kind of) conditional logic and Veltman's update semantics, and define goals in these terms and preferences in particular. Their work is more motivated from the perspective of social science and particularly organisation theory than from our standpoint of specifying and realising an intelligent system.

- $\pi : S \rightarrow (\mathcal{P} \rightarrow \{tt, ff\})$ *is a truth assignment function to the atoms per state;*
- $r, t : \mathcal{L}_{ACT,SA} \rightarrow 2^{S \times S}$ *are state transition relations per action, satisfying the properties of Definition 2.2, as well as some constraints regarding the* `revise` *and* `commit` *actions, to which we will return in the sequel. The* r *transition relation models the transition resulting from performing an action, with respect to the aspect of the agent's opportunity (provided by the environment) to perform the action, regardless of its ability to perform this action, whereas the* t *transition relation models the transition resulting from doing the action, with respect to the aspect of the agent's (internal) capability to perform the action, regardless whether it has the actual opportunity to do so (cf. [21]);*
- $R_{\mathbf{K}}, R_{\mathbf{B}}, R_{\mathbf{D}} \subseteq S \times S$*, such that* $R_{\mathbf{K}}$ *is an equivalence relation,* $R_{\mathbf{B}}$ *is serial, transitive and euclidean, and* $R_{\mathbf{B}} \subseteq R_{\mathbf{K}}$*;*
- $Agenda : S \rightarrow 2^{\mathcal{L}_{ACT}}$ *is a function that yields the agent's agenda of commitments, consisting of the set of actions it is committed to, at any state.*

The interpretation of the (new) formulas in $\mathcal{L}_{SA}$ now reads:

Definition 7.3 *Let* $\mathcal{M} = \langle S, \pi, r, t, R_{\mathbf{K}}, R_{\mathbf{B}}, R_{\mathbf{D}}, Agenda \rangle$ *and* $s \in S$*. Then:*

- $\mathcal{M}, s \models \mathbf{K}\varphi$ *iff* $\mathcal{M}, s' \models \varphi$ *for all* s' *with* $R_{\mathbf{K}}(s, s')$*;*
- $\mathcal{M}, s \models \mathbf{B}\varphi$ *iff* $\mathcal{M}, s' \models \varphi$ *for all* s' *with* $R_{\mathbf{B}}(s, s')$*;*
- $\mathcal{M}, s \models \mathbf{D}\varphi$ *iff* $\mathcal{M}, s' \models \varphi$ *for all* s' *with* $R_{\mathbf{D}}(s, s')$*;*
- $\mathcal{M}, s \models \mathbf{Com}\alpha$ *iff* $\alpha \in Agenda(s)$*;*[8]
- $\mathcal{M}, s \models [\![\alpha]\!]\varphi$ *iff* $\mathcal{M}, s' \models \varphi$ *for all* s' *with* $t(\alpha)(s, s')$*;*
- $\mathcal{M}, s \models \ll \alpha \gg \varphi$ *iff* $\mathcal{M}, s' \models \varphi$ *for some* s' *with* $t(\alpha)(s, s')$*;*

These basic modalities serve as a basis to define further operators, such as ability, opportunity, practical possibility, can, goal, (possible) intend, as follows:

Definition 7.4
- *(ability)* $\mathbf{A}\alpha \equiv \ll \alpha \gg \mathtt{tt}$*, i.e., an agent is able to do an action iff there is a successor state w.r.t. the* t*-relation;*
- *(opportunity)* $\mathbf{O}\alpha \equiv \langle \alpha \rangle \mathtt{tt}$*, i.e., an agent has the opportunity to do an action iff there is a successor state w.r.t. the* r*-relation;*

[8] This is a simplification of our treatment in [37], where we also incorporated semantical implications of such a action into the notion of commitment, dealing e.g. with initial computations of the action.

- *(practical possibility)* $\mathbf{P}(\alpha,\varphi) \equiv \mathbf{A}\alpha \wedge \mathbf{O}\alpha \wedge \langle\alpha\rangle\varphi$, *i.e., an agent has the practical possibility to do an action with result* φ *iff it is both able and has the opportunity to do that action and the result of actually doing that action leads to a state where* φ *holds;*

- *(can)* $\mathbf{Can}(\alpha,\varphi) \equiv \mathbf{KP}(\alpha,\varphi)$, *i.e., an agent can do an action with a certain result iff it knows it has the practical possibilty to do so;*

- *(realisability)* $\Diamond\varphi \equiv \exists a_1,\ldots,a_n\mathbf{P}(a_1;\ldots;a_n,\varphi)$[9], *i.e., a state property* φ *is realisable iff there is a finite sequence of atomic actions of which the agent has the practical possibility to perform it with the result* φ*;*

- *(goal)* $\mathbf{G}\varphi \leftrightarrow \neg\varphi \wedge \mathbf{D}\varphi \wedge \Diamond\varphi$, *i.e., a goal is a formula that is not (yet) satisfied, but desired and realisable.*[10]

- *(possible intend)* $\mathbf{I}(\alpha,\varphi) \equiv \mathbf{Can}(\alpha,\varphi) \wedge \mathbf{KG}\varphi$, *i.e., an agent (possibly) intends an action with a certain result iff the agent can do the action with that result and it moreover knows that this result is a goal of his.*

We are now in a position to state the constraints for the `revise` and `commit` actions. Given a $\mathcal{L}_{SA}$-model $\mathcal{M}$, we define the semantics of these actions as model/state transformers:

1. $r(\texttt{revise}\varphi)(\mathcal{M},s) = \mathit{update\_belief}(\varphi,(\mathcal{M},s))$, and likewise for $t(\texttt{revise}\varphi)$;

2. $r(\texttt{commit}\alpha)(\mathcal{M},s) = \mathit{update\_agenda}(\alpha,(\mathcal{M},s))$, if $\mathcal{M},s \models \mathbf{I}(\alpha,\varphi)$ for some φ, otherwise $r(\texttt{commit}\alpha)(\mathcal{M},s) = \emptyset$ (indicating failure of the commit action), and likewise for $t(\texttt{commit}\alpha)$.

Here *update_belief* and *update_agenda* are functions that update the agent's belief and agenda, respectively. Their formal definitions can be found in [26, 28] and [37]. The `revise` operator can be used to cater for revisions due to observations and communication with other agents, which we will not go into further here (see [28]). Note that the revise and commit action are *'model-transforming'* actions rather than 'state-transforming' ones as is the usual interpretation of actions in dynamic logic. This should not surprise us too much, since these actions change the mental state of the agent, which in our Kripke-style representation involves (part of) the whole model rather than just one (the actual) state.

Besides the familiar properties from epistemic logic (see e.g. [12, 36]), typical properties of this framework, called the KARO logic, include (cf. [26, 37]):

1. $\models \mathbf{O}(\alpha;\beta) \leftrightarrow \langle\alpha\rangle\mathbf{O}\beta$

[9] We abuse our language here slightly, since strictly speaking we do not have quantification in our object language. See [37] for a proper definition.

[10] In fact, here we again simplify matters slightly. In [37] we also stipulate that a goal should be explicitly selected somehow from the desires it has, which is modelled in that paper by means of an additional modal operator. Here we leave this out for simplicity's sake.

2. $\models \mathbf{A}(\alpha;\beta) \leftrightarrow \ll \alpha \gg \mathbf{A}\beta$

3. $\models \mathbf{Can}(\alpha;\beta,\varphi) \leftrightarrow \mathbf{Can}(\alpha,\mathbf{P}(\beta,\varphi))$

4. $\models [\texttt{revise}\varphi]\mathbf{B}\varphi$

5. $\models \mathbf{K}\neg\varphi \leftrightarrow [\texttt{revise}\varphi]\mathbf{B}\texttt{ff}$

6. $\models \mathbf{K}(\varphi \leftrightarrow \psi) \rightarrow ([\texttt{revise}\varphi]\mathbf{B}\chi \leftrightarrow [\texttt{revise}\psi]\mathbf{B}\chi)$

7. $\models \mathbf{I}(\alpha,\varphi) \rightarrow \langle\texttt{commit}\alpha\rangle\mathbf{Com}\alpha$

7.1.2 Actions with Typical Effects

As we have seen before, when we reason about actions we have to take care of an economic specification of the effects and, perhaps even more importantly, the non-effects of actions. The same problem, of course, holds for actions as performed or initiated by agents. As we have also seen, a way to do this is to consider *default* or *typical* effects of actions. In [11] it has been attempted to incorporate this in an agent setting, more in particular in a KARO-like logic. Here we discuss the main idea(s) of their approach.

Firstly to the logical language a new construct $\langle\alpha\rangle\varphi,\psi$ is added with as intended reading that the agent has the opportunity to perform the action α and as a result of this event φ holds (always) and ψ *typically* holds.

Semantically, in the Kripke model, besides the usual accessibility relation r, a relation r° is added that deals with the *typical* results of an action; it is stipulated that $r^\circ(\alpha) \subseteq r(\alpha)$. However, contrary to what one might expect, in the approach of [11] the meaning of the construct $\langle\alpha\rangle\varphi,\psi$ is not given by means of r°. Instead the meaning of this construct is given only in terms of the relation r: $\mathcal{M},s \models \langle\alpha\rangle\varphi,\psi$ iff $\forall s' : r(\alpha)(s,s') \Rightarrow \mathcal{M},s' \models \varphi$ and $\exists s'' : r(\alpha)(s,s'')$ & $\mathcal{M},s'' \not\models \neg\psi$.

The typical effects (and the semantical function r°) come into play in a rather sophisticated way. So-called scenarios are introduced, which more or less consist of a sequence of actions $\alpha = \alpha_1;\ldots;\alpha_n$ together with a pre- and postcondition, γ_{pre} and γ_{post}, respectively. Next a so-called epistemic path is defined as a sequence of states in the model $< s_0,\ldots,s_n >$, such that s_0 satisfies γ_{pre}, s_n satisfies γ_{post}, and the s_i are such that $r(\alpha)(s_i,s_{i+1})$ holds.[11] (In terms of dynamic logic it thus holds in s_0 that $\gamma_{pre} \wedge [\alpha]\gamma_{post}$.) The state s_n is called an end state of the scenario. Now in this path it is counted how many of the s_i are in fact *a*typical, i.e. such that $\neg r^\circ(\alpha)(s_i,s_{i+1})$ holds. If a path has less atypical states than another, it is more preferred. This is used to define a preference-based nonmonotonic inference relation between scenarios and formulas as follows: $SC \mid\!\approx \beta$ (where SC is a scenario) iff for every end state s of the most preferred epistemic paths with respect to the scenario SC it

[11] In this simplified account I leave out the requirements regarding capabilities.

holds that $\mathcal{M}, s \models \beta$. We thus see here a sophisticated mixture of dynamic logic (or rather the KARO logic) and nonmonotonic inference techniques in order to treat typical results / effects of actions.

7.1.3 Real World Agents

When considering agents that operate in the real world, such as robots, also a lot of additional uncertainty comes in. For example, when a robot gets information from sensor devices this may contain errors, both random and systematic ones. The same holds for the other actions that agent performs in the real world. Executing a command of moving ahead for 2 meters may result in actually moving ahead for 1,9 meters. This depends on the devices (e.g. wheels) within the robot that actually achieve such a command but also on the environment (e.g. being on a slippery floor or on a slope). To treat the behaviour of a real world agent properly one has to resort to means of describing such forms of uncertainty in the agent's dynamics. There has been some work on this already [17, 44, 45]. In [52, 53] we are now trying to bring probabilistic reasoning within the framework of dynamic logic. As the work is still in a preliminary stage I restrict myself to an informal description of the main idea. As before we treat actions of the agent as model-transformers. However, these models now contain also probabilistic information. Roughly one can say that the 'state' of a real-world agent is given by a set of possible worlds with a probability distribution, indicating the probability that the agent is in that particular world. Now, for instance an observe action will result in a new 'state' described by a set of possible worlds where possibly some worlds have been eliminated and the probability distribution is adjusted according to Bayesian reasoning. This models the fact that by observation the agent learns something about the state it's in (cf. [52]). Other actions (like a move) are given by a transition to a model in which every possible world is replaced by a set of worlds with a probability distribution on them reflecting the agent's uncertainty (for example, with respect to its position in the case of a move action) after execution of the action [53].

7.2 Multi-Agent Systems

In the agent systems community it is appreciated that probably the most interesting and useful applications of such systems will involve a number of agents (possibly even very many). These systems have great potential in applications such as electronic commerce. To describe, specify and realise such multi-agent systems (MAS) one needs to go beyond the individual level of agents (such as a description of their mental states / attitudes), but one has also to pay serious attention to social attitudes / behaviour of the agents in such a system. Things that naturally come to mind here are communication, co-ordination, negotiation, co-operation and the like. Although it is not entirely clear how much of this can be specified by purely logical means (it is, for instance, quite obvious

that other theoretical concepts and techniques such as stemming from decision and game theory are of crucial importance to obtain an adequate description), it is nevertheless interesting to investigate whether logic (and dynamic logic in particular) is also of some help in this context. Some work in this direction has been done already.

7.2.1 Epistemic Group Updates

For example, in [2] the issue of belief updates within a MAS has been considered. The generalization of belief updates from the single agent case to the multi-agent one is by no means trivial, since one should take care of an adequate representation of whose belief is updated and whose is not! For instance, if agent i communicates to agent j some information such that another agent k is not aware of this, the knowledge/belief of agent k, and in particular his *beliefs about the beliefs of agents i and j (!)*, should remain unaffected. Whereas in the single agent it is sufficient to eliminate 'arrows' in the Kripke model (reflecting the elimination of epistemic possibilities) this is not an adequate procedure any more in the multi-agent case. Baltag *et al.* solve the problem by *adding* arrows instead. The main idea is that, for instance in the previous example, the epistemic relations of agent j change (by eliminating epistemic possibilities), from the perspective of agent k things should remain the same, so that his accessibility relations should still point to worlds where the agents i and j have still the old epistemic possibilities. Precisely this effect can be obtained by copying the old model (for agent k) as well as adding a submodel dealing with agents i and j in isolation.

In [15, 14] related work is reported from a more linguistic perspective, using the framework of Veltman's update semantics [51].[12] Here group updates of the form $U_G\alpha$ are considered, denoting the update of group G by means of action or 'program' α. These programs are either tests $\varphi?$, group updates again, or sequential or alternative compositions of programs. A sound and complete axiomatisation, based on dynamic logic, is given with typical axioms such as:

- $[U_G\alpha]\mathbf{B}_i\varphi \leftrightarrow \mathbf{B}_i\varphi$ if $i \notin G$
- $[U_G\alpha]\mathbf{B}_i\varphi \leftrightarrow \mathbf{B}_i[\alpha][U_G\alpha]\varphi$ if $i \in G$

The former is called the *privacy* axiom, since it expresses that the belief of someone who is not involved in the group update will not change its beliefs. The latter is called the *Ramsey* axiom; it expresses that after a group update with action α an agent believes φ just in case s/he already believed that after executing α an update with $U_G\alpha$ could only result in a world where φ would be true. (This is related to the so-called Ramsey test in philosophical logic.)

[12] Here we must also mention [30], where also similar updates on Kripke models are investigated; however, here these updates are not included in the logical language itself but only considered in the meta-language.

7.2.2 Communication and Speech Acts

Another example of the use of (dynamic) logic for describing MAS is that pertaining to communication. Dignum and Van Linder [8] have attempted to deal with agent communication by means of dynamic logic. The idea here is to view communication between agents as *speech acts*, that is to say actions by means of speech, so that it becomes amenable to a logic for reasoning about actions such as dynamic logic. In fact they use a rather sophisticated mixture of the KARO logic from the previous subsection and (a refinement of) dynamic deontic logic from section 5, since in their set-up some speech acts may give rise to obligations! (E.g. they have an operator $\mathbf{Obl}_{ij}$ such that $\mathbf{Obl}_{ij}\alpha$ expresses that agent i has an obligation to agent j to perform action α.) Furthermore, of course, in a multi-agent setting it must be indicated who is performing the action. We do this here by employing the notation $do_i(\alpha)$, denoting that agent i is performing the action α. They consider the following speech acts: commitments, directions, declarations and assertions, formalised by the actions:

1. $COMMIT(i, j, \alpha)$ denoting the act of commitment of agent i to agent j to perform the action α.
2. $DIR(i, j, \alpha)$ denoting the act of agent i directing agent j to perform action α.
3. $DECL(i, \varphi)$ denoting the act of declaration of i that φ holds.
4. $ASSN(i, j, \varphi)$ denoting the act of agent i asserting to agent j that φ holds.

Without going too much into the details of the rather involved semantic framework, it should come as no surprise that these actions are given a model-transforming interpretation again (like the revise and commit actions for single agents we saw before). Suffice here to say that

1. $COMMIT(i, j, \alpha)$ changes the model in such a way that afterwards there exists an obligation of i to j to perform α, i.e. $\mathbf{Obl}_{ij}$ holds.
2. $DIR(i, j, \alpha)$ changes the model such that afterwards j has an obligation to i to perform α. The action only succeeds if the agent i has the authority to give this directive.
3. $DECL(i, \varphi)$ changes the model such that φ holds in all resulting states, provided that φ is consistent, and the agent i has the authority to perform this declarative action.

(The action $ASSN$ is similar to the multi-agent belief update we have seen before, and I leave it out here.)

Typical formulas that are valid in this framework are thus:

- $[do_i(COMMIT(i,j,\alpha))]\mathbf{Obl}_{ij}\,\alpha$
- $auth(i,DIR(i,j,\alpha)) \rightarrow [do_i(DIR(i,j,\alpha))]\mathbf{Obl}_{ji}\alpha$
- $auth(i,DECL(i,\varphi)) \rightarrow [do_i(DECL(i,\varphi))]\varphi$

7.2.3 Social Attitudes : a Challenge for the Future

Of course, the behaviour of a non-trivial multi-agent system with possibly many heterogeneous agents involves much more than their knowledge and the speech acts between them. Also more complex social attitudes are to be investigated and described. Examples include negotiations, co-ordination of behaviours, co-operation and competition, delegation, trust, coping with joint intentions and goals, but also 'fault-tolerant' behaviour in the sense that if one agent fails to accomplish a task for some reason, another can take over or can at least take some compensatory action. Although already interesting work has been done on these issues (cf. e.g. [4, 7][13]), I believe that there is still a lot to be done here. Obviously a comprehensive and adequate treatment of these matters needs elements from game and decision theory (or theory of economics more in general) as well as the theory of distributed computing. On the other hand, since it is obvious that here, too, (the description of) dynamics plays an important role, there is no *a priori* reason to doubt the use of dynamic logic based on the idea of social actions viewed as transformers of models capturing these social attitudes in this context as well. I think it is a very interesting issue to study where and how these 'extra-logical' elements such as the ones mentioned above may fit in.

8 Conclusion

In this paper I have reviewed a number of different instances of knowledge representation involving agents, actions and changes, in which it has been proven (or at least made plausible) to be fruitful to employ dynamic logic as a basic action logic. From the instances shown I believe the wide applicability of dynamic logic is clearly demonstrated. Of course, in each specific application domain the bare dynmic logic has to be 'embellished' to increase expressibility, but dynamic logic remains the 'core' of the resulting system.

Of course, to reason about the dynamics of (intelligent) systems one might also consider alternatives. A prominent 'rival' of dynamic logic is temporal logic, the logic of time [3], which has also been put to use for the purpose of the verification and specification of programs. Although there are some similarities between dynamic and temporal logic (e.g. both are modal logics, and both can be used to descibe the (temporal) behaviour of systems), a main difference between the two is that in dynamic logic the actions (and perhaps even the actor)

[13] In the latter paper a formalisation of goal formation / generation within a group of agents is presented in a KARO-like logic containing dynamic logic operators

involved are mentioned explicitly in the object language, while this is not the case in temporal logic (although, of course, one may add additional predicates to express aspects of actions). In my opinion both temporal and dynamic logics have their merits, and may be convenient tools for different applications.

References

[1] A.R. Anderson, A Reduction of Deontic Logic to Alethic Modal Logic, *Mind* 67, 1958, pp. 100–103.

[2] A. Baltag, L.S. Moss & S. Solecki, The Logic of Public Announcements, Common Knowledge, and Private Suspicions, in: *Theoretical Aspects of Rationality and Knowledge (Proc. TARK 1998)* (I. Gilboa, ed.), Morgan Kaufmann, San Francisco, 1998, pp. 43–56.

[3] J. van Benthem, Temporal Logic, in: *Handbook of Logic in Artificial Intelligence and Logic Programming, Vol. 4: Epistemic and Temporal Reasoning* (D.M. Gabbay, C.J. Hogger & J.A. Robinson, eds.), Clarendon Press, Oxford, 1995, pp.241–350.

[4] C. Castelfranchi & R. Falcone, Principles of Trust for MAS: Cognitive Anatomy, Social Importance, and Quantification, in: *Proc. 3rd Int. Conf. on Multi-Agent Systems (ICMAS'98)* (Y. Demazeau, ed.), IEEE, Los Alamitos, CA, 1998, pp. 72–79.

[5] J. Coenen, Top-Down Development of Layered Fault-Tolerant Systems and Its Problems - a Deontic Perspective, *Annals of Mathematics and Artificial Intelligence* 9(1,2), 1993, pp.133–150.

[6] P. Cousot, Methods and Logics for Proving Programs, in: J. van Leeuwen (ed.), *Handbook of Theoretical Computer Science, Vol. B: Formal Models and Semantics*, Elsevier, Amsterdam, 1990, pp. 841–993.

[7] F. Dignum & R. Conte, Intentional Agents and Goal Formation, in: Intelligent Agents IV (M.P. Singh, A. Rao & M.J. Wooldridge, eds.), Springer, Berlin, 1998, pp. 231–243.

[8] F. Dignum & B. van Linder, Modelling Social Agents: Communication as Action, in: *Intelligent Agents III* (J.P. Müller, M.J. Wooldridge & N.R. Jennings, eds.), Springer, Berlin, 1997, pp. 205–218.

[9] F.P.M. Dignum, J.-J. Ch. Meyer & R.J. Wieringa, Free Choice and Contextually Permitted Actions, *Studia Logica* 57(1), 1996, pp. 193-220.

[10] F. Dignum, J.-J. Ch. Meyer, R.J. Wieringa & R. Kuiper, A Modal Approach to Intentions, Commitments and Obligations: Intention plus Commitment Yields Obligation, in: *Deontic Logic, Agency and Normative Systems (Proc. DEON'96)* (M.A. Brown & J. Carmo, eds.), Workshops in Computing, Springer, Berlin, 1996, pp. 80–97.

[11] B. Dunin-Keplicz & A. Radzikowska, Epistemic Approach to Actions with Typical Effects, in: *Symbolic and Quantitative Approaches to Reasoning and Uncertainty (Proc. ECSQARU'95)* (Chr. Froideveaux & J. Kohlas, eds.), Lecture Notes in Artificial Intelligence 946, Springer, Berlin, 1995, pp. 180–188.

[12] R. Fagin, J.Y. Halpern, Y. Moses & M. Vardi, *Reasoning about Knowledge*, The MIT Press, Cambridge, MA, 1995.

[13] M. Fischer & R. Ladner, Propositional Dynamic Logic of Regular Programs, *J. Comput. System Sci.* 18, 1979, pp. 194–211.

[14] J. Gerbrandy, Bisimulations on Planet Kripke, Ph.D. Thesis, University of Amsterdam, 1999.

[15] J. Gerbrandy & W. Groeneveld, Reasoning about Information Change, *Journal of Logic, Language, and Information* 6, 1997, pp. 147–169.

[16] J. Groenendijk & M. Stokhof, Dynamic Predicate Logic, *Linguistics and Philosophy* 14(1), 1991, pp. 31–100.

[17] J.Y. Halpern, A Logical Approach to Reasoning about Uncertainty: a Tutorial, in: *Discourse, Information and Communication* (X. Arrazola, K. Korta & F.J. Pelletier, eds.), Kluwer, 1997.

[18] D. Harel, Dynamic Logic, in: D. Gabbay & F. Guenthner (eds.), *Handbook of Philosophical Logic, Vol. II*, Reidel, Dordrecht/Boston, 1984, pp. 497–604.

[19] D. Harel, *First-Order Dynamic Logic*, Lectures Notes in Computer Science 68, Springer, Berlin, 1979.

[20] C.A.R. Hoare, An Axiomatic Basis for Computer Programming, *Comm. ACM* 12, 1969, pp. 576–580.

[21] W. van der Hoek, J.-J. Ch. Meyer & J.W. van Schagen, Formalizing Potential of Agents – The KARO Framework Revisited, in: *Proc. of the 7th CSLI Workshop on Logic, Language and Computation* (M. Faller, S. Kaufmann M. Pauly, eds.), CSLI Publications, Stanford, 1999, to appear.

[22] Z. Huang, M. Masuch & L. Pólos, ALX, an Action Logic for Agents with Bounded Rationality, *Artificial Intelligence* 82, 1996, pp. 75–127.

[23] N.R. Jennings, K. Sycara & M. Wooldridge, A Roadmap of Agent Research and Development, *Autonomous Agents and Multi-Agent Systems* 1, 1998, pp. 7–38.

[24] G.N. Kartha & V. Lifschitz, Actions with Indirect Effects (Preliminary Report), in: *Proc. KR'94*, 1994, pp. 341–350.

[25] D. Kozen & J. Tiuryn, Logics of Programs, in: J. van Leeuwen (ed.), *Handbook of Theoretical Computer Science, Vol. B: Formal Models and Semantics*, Elsevier, Amsterdam, 1990, pp.789–840.

[26] B. van Linder, W. van der Hoek & J.-J. Ch. Meyer, Actions that Make You Change Your Mind: Belief Revision in an Agent-Oriented Setting, in: *Knowledge and Belief in Philosophy and Artificial Intelligence* (A. Laux & H. Wansing, eds.), Akademie Verlag, Berlin, 1995, pp. 103–146.

[27] B. van Linder, W. van der Hoek & J.-J. Ch. Meyer, Formalising Motivational Attitudes of Agents: On Preferences, Goals and Commitments, in: Intelligent Agents Volume II – Agent Theories, Architectures, and Languages (M. Wooldridge, J.P. Müller & M. Tambe, eds.), Lecture Notes in Computer Sciences (Subseries LNAI) 1037, Springer-Verlag, 1996, pp. 17–32.

[28] B. van Linder, W. van der Hoek & J.-J. Ch. Meyer, Seeing is Believing (And So Are Hearing and Jumping), *Journal of Logic, Language and Information* 6, 1997, pp. 33–61.

[29] B. van Linder, W. van der Hoek & J.-J. Ch. Meyer, Formalising Abilities and Opportunities of Agents, *Fundamenta Informaticae* 34(1, 2), 1998, pp. 53–101.

[30] A. Lomuscio, Knowledge Sharing among Ideal Agents, Ph.D. Thesis, University of Birmingham, 1999.

[31] W. Lukaszewicz and E. Madaliǹska-Bugaj, Reasoning about Action and Change using Dijkstra's semantics for Programming Languages: Preliminary Report, in *Proc. IJCAI-95*, Montreal, 1995, pp. 1950–1955.

[32] E. Mally, *Grundgesetze des Sollens, Elemente der Logik des Willens*, Leuschner & Lubensky, Graz, 1926.

[33] R. van der Meyden, The Dynamic Logic of Permission, *Proc. 5th IEEE Conf. on Logic in Computer Science*, Philadephia, 1990, pp. 72–78.

[34] J.-J. Ch. Meyer, A different approach to deontic logic: Deontic logic viewed as a variant of dynamic logic, *Notre Dame Journal of Formal Logic*, vol.29, pp. 109–136, 1988.

[35] J.-J. Ch. Meyer & P. Doherty, Preferential Action Semantics (Preliminary Report), in: *Formal Models of Agents* (J.-J. Ch. Meyer & P.Y. Schobbens, eds.), Springer, to appear.

[36] J.-J. Ch. Meyer & W. van der Hoek, *Epistemic Logic for AI and Computer Science*, Cambridge University Press, 1995.

[37] J.-J. Ch. Meyer, W. van der Hoek & B. van Linder, A Logical Approach to the Dynamics of Commitments, 1999, to appear in AI Journal.

[38] J.-J. Ch. Meyer, R.J. Wieringa & F.P.M. Dignum, The Role of Deontic Logic in the Specification of Information Systems, in: Logics for Databases and Information Systems (J. Chomicki & G. Saake, eds.), Kluwer, Boston/Dordrecht, 1998, pp. 71–115.

[39] R.C. Moore, A Formal Theory of Knowledge and Action, in: J.R. Hobbs & R.C. Moore (eds.), *Formal Theories of the Commonsense World*, Ablex, Norwood NJ, 1985, pp. 319–358.

[40] V. Pratt, Semantical Considerations on Floyd-Hoare Logic, in Proc. 17th IEEE Symp. on Foundations of Computer Science, 1976, pp. 109–121.

[41] L.M.M. Royakkers, *Extending Deontic Logic for the Formalisation of Legal Rules*, Kluwer Academic Publishers, Dordrecht/Boston, 1998.

[42] E. Sandewall, *Features and Fluents: A Systematic Approach to the Representation of Knowledge about Dynamical Systems*, Oxford University Press, Oxfors, 1994.

[43] E. Sandewall & Y. Shoham, Nonmonotonic Temporal Reasoning, in: *Handbook of Logic in Artificial Intelligence and Logic Programming Vol.4 (Epistemic and Temporal Reasoning)* (D.M. Gabbay, C.J. Hogger & J.A. Robinson, eds.), Oxford University Press, Oxford, 1994.

[44] M.P. Shanahan, Moise and the Common Sense Informatic Situation for a Mobile Robot, *Proc. AAAI'96*, 1996, pp. 1098–1103.

[45] M.P. Shanahan, Robotics and the Common Sense Informatic Situation, *Proc. ECAI'96*, 1996, pp. 684–688.

[46] C. Sierra, L. Godo, R. López de Màntaras & M. Manzano, Descriptive Dynamic Logic and Its Application to Reflective Architectures, *Future Generation Computer Systems* 12, 1996, pp. 157–171.

[47] A. Soeteman, *Logic in Law*, Kluwer Academic Publishers, Dordrecht/Boston, 1989.

[48] P.A. Spruit, R.J. Wieringa & J.-J. Ch. Meyer, Axiomatization, Declarative Semantics and Operational Semantics of Passive and Active Updates in Logic Databases, *Journal of Logic and Computation* 5(1), 1995, pp. 27-70.

[49] P. Spruit, R.J. Wieringa & J.-J. Ch. Meyer, Regular Database Update Logics, 1999, submitted.

[50] C. Stirling, Modal and Temporal Logics, in: S. Abramsky, D.M. Gabbay & T.S.E. Maibaum (eds.), *Handbook of Logic in Computer Science, Vol. II*, Carendon Press, Oxford, 1992, pp. 477–563.

[51] F. Veltman, Defaults in Update Semantics, *Journal of Philosophical Logic* 25, 1996, pp. 221–261.

[52] M. de Weerdt, F.S. de Boer, W. van der Hoek & J.-J. Ch. Meyer, Imprecise Observations of Mobile Robots Specified by a Modal Logic, in: *Proc. of the fifth annual conference of the Advanced School for Computing and Imaging (ASCI'99)*, (M. Boasson, J.A. Kaandorp, J.F.M. Tonino & M.G. Vosselman, eds.), Heijen, The Netherlands, June 15-17, 1999, pp. 184-190.

[53] M. de Weerdt, F.S. de Boer, W. van der Hoek & J.-J. Ch. Meyer, Specifying Uncertainty of Mobile Robots by Means of a Modal Logic, in preparation.

[54] G.H. von Wright, Deontic Logic, *Mind* 60, 1951, pp. 1–15.

[55] G.H. von Wright, A New System of Deontic Logic, *Danish Yearbook of Philosophy* 1, 1964, pp. 173–182.

[56] M. Wooldridge & N. Jennings, Intelligent Agents: Theory and Practice, *The Knowledge Engineering Review* 10(2), 1995, pp. 115–152.