Formal Semantics of Meta-Level Architectures:
Dynamic Control of Reasoning

Jan Treur

Vrije Universiteit Amsterdam, Department of Artificial Intelligence
De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands
Email: treur@cs.vu.nl
URL: http://www.cs.vu.nl/~treur

Abstract

Meta-level architectures for dynamic control of reasoning processes are quite powerful. In the literature many applications in reasoning systems modelling complex tasks are described, usually in a procedural manner. In this paper we present a semantic framework based on temporal partial logic to describe the dynamics of reasoning behaviour. Using these models the semantics of the behaviour of the whole (meta-level) reasoning system can be described by a set of (intended) temporal models.

1 Introduction

In the literature on meta-level architectures and reflection (e.g., [28]) two separate streams can be distinguished: a logical stream (e.g., [4], [19], [37]) and a procedural stream (e.g., [10], [11]). Unfortunately there is a serious gap between the two streams. In the logical stream one restricts oneself often to *static reflections*; i.e., of facts the truth of which does not change during the reasoning: e.g., **provable**(**A**) (with **A** an object-level formula). In the procedural stream usually facts are reflected the truth of which changes during the whole reasoning pattern; e.g. control statements like **current_goal**(**A**), with **A** an object-level formula, that are sometimes true and sometimes false during the reasoning. If applications to dynamic control of complex reasoning tasks are concerned these *dynamic reflections* are much more powerful (for applications see, e.g.: [11], [10], or [6], [7], [8], [9], [18], [30], [31], [33], [35]). However, a logical basis has not been investigated in depth. The current paper provides such a logical foundation (based on temporal logic) of meta-level architectures for dynamic control. The semantical framework allows for the analysis of these dynamic meta-level architectures by logical means. It can be viewed as a contribution to bridge the gap between the logical stream and the procedural stream.

A meta-level architecture consists of two interacting components that reason at different levels: the object-level component and the meta-level component. The interactions between the components are: upward reflection (information transfer from object-level component to meta-level component) and downward reflection (information transfer from meta-level component to object-level component). In a

meta-level architecture each of the reasoning processes in one of the components can be assigned its own *local semantics* (local in view of the whole system) that can be formally described according to well-known approaches to static (declarative) and dynamic (procedural) semantics (for instance as known from logic programming). In this local semantics a *static view* (the contents; declarative) and a *dynamic view* (the control; procedural) can be distinguished, and these views can be treated (to a certain extent) as orthogonal (e.g., see [27]).

The crucial point of a meta-level architecture for dynamic control is that the semantics of the meta-level component relates in some manner to (the control of) the reasoning process of the object-level component. To obtain an *overall semantics* for the whole system, the crux is to formally describe the precise *semantic connection* between the two components. The semantic connection is used in a bidirectional manner. Firstly, meta-level reasoning *is about* (or refers to) process aspects of the reasoning of the object-level component and uses information about that (via upward reflection). Secondly, the results of its reasoning may *affect* the control of this object-level reasoning process by changing its control settings (downward reflection). Therefore meta-level architectures enable one to represent control knowledge in the system in an explicit declarative manner.

A formal semantic connection between the two components should relate some (formal) description of procedural (inference process control) aspects of the object-level component to the formal declarative description of the meta-level component. Therefore for the overall semantics of this type of reasoning system a global distinction between a static view and a dynamic view essentially cannot be made (as an extension of the distinction that can be made within each of the components). The views are not orthogonal in this case: to a certain extent they are defined in terms of each other. In particular, it is impossible to provide independent declarative semantics for such systems without taking into account the dynamics (of the object-level component). For the overall architecture a formal semantical description is needed that systematically integrates both views. The lack of such overall semantics for complex reasoning systems (with meta-level reasoning capabilities) was one of the major open problems that were identified in [36].

In this paper we develop a formal framework where partial models are used to explicitly represent (current) information states (see also [24]). This enables us to represent inference processes within each of the components as transitions between partial models, and a trace of a reasoning process as a partial temporal model. First, in Section 2 some basic notions from Temporal Partial Logic are introduced. Next, in Section 3 we give a formalization of a static view and a dynamic view on the object-level reasoning component. In Section 4 a temporal interpretation of meta-level reasoning is introduced and in Section 5 formal semantics for reasoning patterns of meta-level architectures for dynamic control are presented based on partial temporal models that formalize the overall reasoning traces. Finally, in Section 6 an example is presented.

2 Basic notions of Temporal Partial Logic

In this section we will introduce the formal notions of partial logic and temporal partial logic that are needed later on.

Definition 2.1 (signature and propositional formula)
a) A *signature* Σ is an ordered sequence of (propositional) atom names, or a sequence of sort, constant, function and predicate symbols in (many-sorted) predicate logic. By $\Sigma_1 \subseteq \Sigma_2$ we denote that Σ_1 is a *subsignature* of Σ_2. The *disjoint union* of two signatures Σ_1 and Σ_2 is denoted by $\Sigma_1 \oplus \Sigma_2$. A *mapping of signatures* $\alpha : \Sigma_1 \to \Sigma_2$ is a mapping from the set of symbols of Σ_1 into the set of symbols of Σ_2 such that sorts are mapped to sorts, constants to constants, predicates to predicates, functions to functions, and the arities and argument-sort relations are respected.
b) The set **At(Σ)** is the *set of (ground) atoms* based on Σ. By a (ground) *formula* of signature Σ we mean a proposition built from (ground) atoms using the connectives $\wedge, \to, \neg$. We will call these formulae *propositional formulae*, in contrast to the temporal formulae introduced later on. Below we will assume all formulas ground (closed and without quantifiers). For a finite set **F** of formulae, **con(F)** denotes the conjunction of the elements of **F**; in case **F** is the empty set then by definition **con(F)** is **true**.
By **Lit(Σ)** we denote the *set of ground literals* of signature Σ.

As discussed in [24], partial models can be used to represent information states in a reasoning system; therefore we define:

Definition 2.2 (partial models as information states)
a) An *information state* or *partial model* **M** for the signature Σ is an assignment of a truth value from **{0, 1, u}** to each of the atoms of Σ, i.e. **M: At(Σ) $\to$ {0, 1, u}**. An atom **a** is *true* in **M** if **1** is assigned to it, and *false* if **0** is assigned; else it is called *undefined* or *unknown*. A literal **L** is called *true* in **M**, denoted by $\mathbf{M} \vDash^+ \mathbf{L}$ (resp. *false* in **M**, denoted by $\mathbf{M} \vDash^- \mathbf{L}$) if **M(L) = 1** (resp. **M(L) = 0**) if **L** is an atom and **M(a) = 0** (resp. **M(a) = 1**) if **L = $\neg$ a** with **a $\in$ At(Σ)**. By **Lit(M)** we denote the *set of literals* which are true in **M**.
We call a partial model **M** *complete* if no **M(a)** equals **u** for any **a $\in$ At(Σ)**.
b) The *set of all information states* for Σ is denoted by **IS(Σ)**. If $\Sigma_1 \subseteq \Sigma_2$ then this induces an embedding of **IS(Σ_1)** into **IS(Σ_2)**; we will identify **IS(Σ_1)** with its image under this embedding: **IS(Σ_1) $\subseteq$ IS(Σ_2)**. Furthermore, **IS($\Sigma_1 \oplus \Sigma_2$)** can (and will) be identified with the Cartesian product **IS(Σ_1) x IS(Σ_2)**.
c) We call **N** a *refinement* of **M**, denoted by **M $\leq$ N**, if for all atoms **a $\in$ At(Σ)** it holds: **M(a) $\leq$ N(a)** where the partial order on truth values is defined by
$$\mathbf{u \leq 0, \ u \leq 1, \ u \leq u, \ 0 \leq 0, \ 1 \leq 1}.$$
d) If **K** is a set of formulae of signature Σ, a complete model **M** of signature Σ is called *a model of* **K** if all formulae of **K** are true in **M**. An information state is

consistent with **K** if it can be refined to a complete model that is a model of **K**. By $\mathbf{IS_K(\Sigma)}$ we denote the set of all information states for Σ that are consistent with **K**.

e) If **M** is a partial model for the signature Σ and $\mathbf{S} \subseteq \mathbf{At(\Sigma)}$, then by **M|S** we denote the *restriction* or *reduct* of **M** to **S**, defined by

$$\mathbf{M|S(a)} \quad = \quad \mathbf{M(a)} \quad \text{if } \mathbf{a} \in \mathbf{S}$$
$$\mathbf{u} \quad \text{otherwise (i.e., if } \mathbf{a} \in \mathbf{At(\Sigma)\backslash S})$$

If $\mathbf{S} = \mathbf{At(\Sigma')}$ for some subsignature Σ' of Σ, then we denote **M|S** by **M|Σ'**.

Notice that for partial models **M, N** for Σ it holds $\mathbf{M} \leq \mathbf{N}$ if and only if $\mathbf{M} \vDash^+ \mathbf{L} \Rightarrow \mathbf{N} \vDash^+ \mathbf{L}$ for all literals $\mathbf{L} \in \mathbf{Lit(\Sigma)}$. We base the interpretation of propositional formulae on the Strong Kleene truth tables for the logical connectives (see also Definition 2.6 below); more details and possibilities of partial semantics can be found in [3], [25].

Definition 2.3 (labeled flow of time)
Let *L* be a set of labels.

a) A *(discrete) labeled flow of time* , labeled by *L* is a pair $\mathbf{T} = (\mathbf{T}, (<_i)_{i \in L})$ consisting of a nonempty set **T** of time points and a collection of binary relations $<_i$ on **T**. Here for **s, t** in **T** and $\mathbf{i} \in L$ the expression $\mathbf{s} <_i \mathbf{t}$ denotes that **t** is a (immediate) *successor* of **s** with respect to an arc labeled by **i**. Sometimes we use just the binary relation $\mathbf{s} < \mathbf{t}$ denoting that $\mathbf{s} <_i \mathbf{t}$ for some **i** (for some label **i** they are connected). Thus $<$ is defined as $\cup_i <_i$. We will assume that this relation $<$ is irreflexive, antisymmetric and antitransitive.

We also use the (irreflexive) transitive closure **«** of this binary relation, defined as $<^+$.

b) We call **T** *linear* if **«** is a linear ordering and *rooted* with root **r** if **r** is a (unique) least element: for all **t** it holds $\mathbf{r} = \mathbf{t}$ or $\mathbf{r} \ll \mathbf{t}$. We say **T** satisfies *successor existence* if every time point has at least one successor: for all $\mathbf{s} \in \mathbf{T}$ there exists a $\mathbf{t} \in \mathbf{T}$ such that $\mathbf{s} < \mathbf{t}$.

Definition 2.4 (partial temporal model)
Let Σ be a signature.

a) A *labeled (linear time) partial temporal model* of signature Σ with labeled flow of time **T** is a mapping

$$\mathbf{M: T} \rightarrow \mathbf{IS(\Sigma)}$$

For any fixed time point **t** the partial model **M(t)** is also denoted by $\mathbf{M_t}$; the model **M** can also be denoted by $(\mathbf{M_t})_{\mathbf{t} \in \mathbf{T}}$. If **a** is an atom, and **t** is a time point in **T**, and $\mathbf{M_t(a) = 1}$, then we say in this model **M** *at time point* **t** *the atom* **a** *is true*. Similarly we say that *at time point* **t** *the atom* **a** is *false*, respectively *unknown*, if $\mathbf{M_t(a) = 0}$, respectively $\mathbf{M_t(a) = u}$.

b) The *refinement relation* $\leq$ between partial temporal models is defined as: $\mathbf{M} \leq \mathbf{N}$ if **M** and **N** have the same flow of time and for all time points **t** and atoms **a** it holds $\mathbf{M_t(a)} \leq \mathbf{N_t(a)}$.

c) **M** is called *conservative* if for all $\mathbf{s, t} \in \mathbf{T}$ with $\mathbf{s} < \mathbf{t}$ it holds $\mathbf{M_s} \leq \mathbf{M_t}$.

M _t_

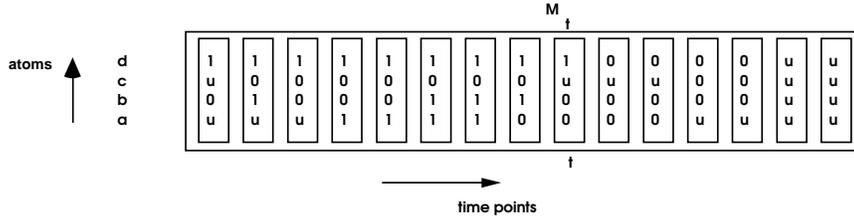| atoms															
d	1	1	1	1	1	1	1	1	1	0	0	0	0	u	u
c	u	0	0	0	0	0	0	0	u	u	u	u	0	u	u
b	0	1	0	0	0	1	1	1	0	0	0	0	0	u	u
a	u	u	u	1	1	1	1	0	0	0	0	u	u	u	u

_t_

time points

Fig 1 Example of a partial temporal model

From now on we will assume that all used labeled flows of time are linear, rooted and satisfy successor existence. This is equivalent to **T** being order-isomorphic to the natural numbers **N**. Therefore in the rest of the paper we will use **N** as our flow of time.

We introduce three temporal operators, **X**, **P** and **C**, referring to the *next* information state, *past* information states and the *current* information state, respectively. Intuitively, the temporal formula **X**α is true at time **t** means that viewed from time point **t**, the formula α is true in the next information state. We use labeled next operators to be able to distinguish different types of steps. The temporal formula **P**α is true at time **t** means that α is true in some past information state. Furthermore we will need an operator that expresses the fact that *currently* α is true (in the *current* information state); this will be the operator **C**. Definition 2.5 makes this formal. Notice that sometimes we will denote the application of the temporal operators like **F**(α); if no confusion is expected, for shortness we write **F**α. We will not need nested operators in this paper, although it would certainly be possible to use them.

Definition 2.5 (semantics of the temporal operators)
Let a propositional formula α, a labeled partial temporal model **M**, a label **i** $\in$ *L* and a time point **t** $\in$ N be given. Then:

a) $(\mathbf{M, t}) \vDash^{+} \mathbf{X_i}\alpha$ $\Leftrightarrow$ $\exists \, \mathbf{s} \in \mathbf{N} \, [\, \mathbf{t} <_\mathbf{i} \mathbf{s} \, \& \, (\mathbf{M, s}) \vDash^{+} \alpha \,]$

         $(\mathbf{M, t}) \vDash^{-} \mathbf{X_i}\alpha$ $\Leftrightarrow$ $(\mathbf{M, t}) \nvDash^{+} \mathbf{X_i}\alpha$

b) $(\mathbf{M, t}) \vDash^{+} \mathbf{C}\alpha$ $\Leftrightarrow$ $(\mathbf{M, t}) \vDash^{+} \alpha$

         $(\mathbf{M, t}) \vDash^{-} \mathbf{C}\alpha$ $\Leftrightarrow$ $(\mathbf{M, t}) \nvDash^{+} \mathbf{C}\alpha$

c) $(\mathbf{M, t}) \vDash^{+} \mathbf{P}\alpha$ $\Leftrightarrow$ $\exists \, \mathbf{s} \in \mathbf{N} \, [\, \mathbf{s} \ll \mathbf{t} \, \& \, (\mathbf{M, s}) \vDash^{+} \alpha \,]$

         $(\mathbf{M, t}) \vDash^{-} \mathbf{P}\alpha$ $\Leftrightarrow$ $(\mathbf{M, t}) \nvDash^{+} \mathbf{P}\alpha$

Now we can make new formulae using conjunctions, negations and implications of these temporal formulae. From now on the word (temporal) formula will be used to denote a formula possibly containing any of the new operators, unless stated otherwise.

5

As we do not need nesting of temporal operators, for convenience we will only consider non-nested formulae.

Definition 2.6 (temporal formulae and their interpretation)
Let Σ be a signature, let $\mathbf{M}$ be a labeled partial temporal model for Σ, and $\mathbf{t} \in \mathbf{N}$ a time point.
a) A *temporal atom* of signature Σ is a formula $\mathbf{O}\alpha$ where $\mathbf{O}$ is one of the temporal operators in Definition 2.5 and α a propositional formula of signature Σ.
A *temporal formula* of signature Σ is a formula built from temporal atoms of signature Σ, using the logical connectives $\neg, \wedge, \rightarrow$.
b) Any propositional atom $\mathbf{p} \in \mathbf{At}(\Sigma)$ is interpreted according to:

$$(\mathbf{M, t}) \vDash^+ \mathbf{p} \qquad \Leftrightarrow \qquad \mathbf{M(t)(p) = 1}$$
$$(\mathbf{M, t}) \vDash^- \mathbf{p} \qquad \Leftrightarrow \qquad \mathbf{M(t)(p) = 0}$$

For the interpretation of a temporal atom, see Definition 2.5.
c) For any two temporal or propositional formulae φ and ψ:

  (i) $(\mathbf{M, t}) \vDash^+ \varphi \wedge \psi \quad \Leftrightarrow \quad (\mathbf{M, t}) \vDash^+ \varphi \quad$ and $(\mathbf{M, t}) \vDash^+ \psi$

          $(\mathbf{M, t}) \vDash^- \varphi \wedge \psi \quad \Leftrightarrow \quad (\mathbf{M, t}) \vDash^- \varphi \quad$ or $\quad (\mathbf{M, t}) \vDash^- \psi$

  (ii) $(\mathbf{M, t}) \vDash^+ \varphi \rightarrow \psi \Leftrightarrow \quad (\mathbf{M, t}) \vDash^- \varphi \quad$ or $\quad (\mathbf{M, t}) \vDash^+ \psi$

          $(\mathbf{M, t}) \vDash^- \varphi \rightarrow \psi \Leftrightarrow \quad (\mathbf{M, t}) \vDash^+ \varphi \quad$ and $(\mathbf{M, t}) \vDash^- \psi$

  (iii) $(\mathbf{M, t}) \vDash^+ \neg \varphi \qquad\qquad \Leftrightarrow \qquad (\mathbf{M, t}) \vDash^- \varphi$

          $(\mathbf{M, t}) \vDash^- \neg \varphi \qquad\qquad \Leftrightarrow \qquad (\mathbf{M, t}) \vDash^+ \varphi$

d) For any temporal or propositional formula φ:

    $(\mathbf{M, t}) \nvDash^+ \varphi \qquad\qquad \Leftrightarrow \qquad (\mathbf{M, t}) \vDash^+ \varphi \quad$ does not hold

    $(\mathbf{M, t}) \nvDash^- \varphi \qquad\qquad \Leftrightarrow \qquad (\mathbf{M, t}) \vDash^- \varphi \quad$ does not hold

    $(\mathbf{M, t}) \vDash^{\mathbf{u}} \varphi \qquad\qquad \Leftrightarrow \qquad (\mathbf{M, t}) \nvDash^+ \varphi \quad$ and $\quad (\mathbf{M, t}) \nvDash^- \varphi$

e) For a partial model $\mathbf{M}$ and a set of formulae $\mathbf{K}$, by $\mathbf{M} \vDash^+ \mathbf{K}$ we mean $\mathbf{M} \vDash^+ \varphi$ for all $\varphi \in \mathbf{K}$. By $\mathbf{M} \vDash^+ \varphi$ we mean $(\mathbf{M, t}) \vDash^+ \varphi$ for all $\mathbf{t} \in \mathbf{N}$ and by $\mathbf{M}$ is a *temporal model of* $\mathbf{K}$, denoted $\mathbf{M} \vDash^+ \mathbf{K}$, we mean $\mathbf{M} \vDash^+ \varphi$ for all $\varphi \in \mathbf{K}$, where $\mathbf{K}$ is a set of temporal or propositional formulae. A model $\mathbf{M}$ of $\mathbf{K}$ is called a *minimal model* of $\mathbf{K}$ if for any model $\mathbf{M'}$ of $\mathbf{K}$ with $\mathbf{M'} \leq \mathbf{M}$ it holds $\mathbf{M'} = \mathbf{M}$.

    The temporal approach provides declarative semantics for systems that behave dynamically, essentially since time has been put into the domain of consideration in an explicit manner: one reasons both on world states and the time points on which they occur. This means that non-conservative changes in truth values of a statement $\mathbf{b}$ referring to a changing world state are accounted for by considering the statement in fact as two (or more) statements: one $(\mathbf{t, b})$ referring to one time point $\mathbf{t}$, and another one $(\mathbf{s, b})$ referring to another time point $\mathbf{s}$. The truth values of these two statements do not change; e.g., it will always remain true that at time point $\mathbf{t}$ the statement $\mathbf{b}$ holds. Thus a dynamic system is described in a declarative manner. Its set of intended models can be constructed in the temporal sense described above. One specific behaviour of the system corresponds to one of these temporal models. We will work out this general idea for the case of a meta-level architecture. More details on temporal logic can be found in [2], [20].

3 Static and dynamic view on the object-level reasoning

In this section we use the notion of a *partial model* to formalize the information state of the object-level reasoning component at a certain moment. A transition of one information state to another one can be formally described by a *mapping between the partial models* specifying the information states. In a reasoning process such a transition is induced by a reasoning step where a knowledge unit **K** (e.g., a set of rules or a single rule) is used to derive some additional conclusions. The *dynamic interpretation* of such a knowledge unit **K** can be defined as the mapping in the set of all relevant partial models induced by **K**.

Note that information states are defined in terms of literals. This implies that in principle only literal conclusions count in inferences. Therefore we can take advantage of the fact that inference relations, restricted to literal conclusions, that are sound with respect to the classical Tarski semantics are also sound with respect to the partial Strong Kleene semantics (and vice versa), as has been established in [29] (cf. Theorem 2.3, p. 464). In the sequel by $\vdash$ we will denote any sound inference relation that is not necessarily complete (e.g, one of: natural deduction, chaining, full resolution, SLD- resolution, unit resolution, etc.).

3.1 The static view on the object-level reasoning

In this subsection we define the underlying language, logical theory and inference relation of the object-level component. Moreover we define the notions of deductive and semantic closure.

Definition 3.1 (static view on the object-level component)
The *static view on the object-level reasoning component* is a tuple
$$\langle \Sigma_0, \mathbf{OT}, \vdash \rangle$$
with

Σ_0	a signature, called the object-signature
OT	a set of propositional ground formulae expressed in terms of the object-signature
$\vdash$	a classical inference relation (assumed sound but not necessarily complete)

Notice that a literal formula is true in a partial model **M** if and only if according to the classical semantics the formula is true in every complete refinement of **M**.

Definition 3.2 (deductive and semantic closure)
Let **K** be a set of propositional formulae of signature Σ and $\mathrel{|\!\sim}$ a (sound) inference relation or (semantic) entailment relation.

a) For $\mathbf{M} \in \mathbf{IS_K}(\Sigma)$ we define the partial model $\mathbf{cl_K}^{\mathrel{|\!\sim}} (\mathbf{M})$ by

$$\mathbf{cl_K}^{\mid\sim} (\mathbf{M}) \vDash^+ \mathbf{L} \quad \Leftrightarrow \quad \mathbf{K} \cup \mathbf{Lit(M)} \mid\sim \mathbf{L}$$

for any literal **L**. This model is called the *closure* of **M** under **K** (with respect to $\mid\sim$). We call **M** *closed* under **K** (with respect to $\mid\sim$) if **M** = $\mathbf{cl_K}^{\mid\sim}(\mathbf{M})$, or, equivalently, if

$$\mathbf{M} \vDash^+ \mathbf{L} \quad \Leftrightarrow \quad \mathbf{K} \cup \mathbf{Lit(M)} \mid\sim \mathbf{L}$$

b) If $\mid\sim$ is an inference relation $\vdash$ we denote $\mathbf{cl_K}^{\mid\sim}(\mathbf{M})$ by $\mathbf{dc_K}^{\vdash}(\mathbf{M})$ and call it the *deductive closure of* **M** under **K** (with respect to $\vdash$). We call **M** *deductively closed* under **K** if

$$\mathbf{K} \cup \mathbf{Lit(M)} \vdash \mathbf{L} \quad \Leftrightarrow \quad \mathbf{M} \vDash^+ \mathbf{L}$$

i.e., if it is its own deductive closure under **K**.

c) For the classical semantic consequence relation $\vDash$ (based on complete models) we denote $\mathbf{cl_K}^{\vDash}(\mathbf{M})$ by $\mathbf{sc_K(M)}$ and call it the *semantic closure of* **M**. We call **M** *semantically closed* under **K** if

$$\mathbf{K} \cup \mathbf{Lit(M)} \vDash \mathbf{L} \quad \Leftrightarrow \quad \mathbf{M} \vDash^+ \mathbf{L}$$

i.e., if it is its own semantic closure under **K**.

Definition 3.3 (conservation, monotonicity, idempotency)
Let **K** be a set of propositional formulae of signature Σ.
We call the mapping $\alpha : \mathbf{IS_K}(\Sigma) \rightarrow \mathbf{IS_K}(\Sigma)$:

(i) *conservative*   if $\mathbf{M} \leq \alpha(\mathbf{M})$ for all $\mathbf{M} \in \mathbf{IS_K}(\Sigma)$

(ii) *monotonic*   if $\alpha(\mathbf{M}) \leq \alpha(\mathbf{N})$ for all $\mathbf{M}, \mathbf{N} \in \mathbf{IS_K}(\Sigma)$ with $\mathbf{M} \leq \mathbf{N}$

(iii) *idempotent*   if $\alpha(\alpha(\mathbf{M})) = \alpha(\mathbf{M})$ for all $\mathbf{M} \in \mathbf{IS_K}(\Sigma)$

For more properties of this type of functionality mapping, see [34].

Proposition 3.4

Let **K** be a set of propositional formulae of signature Σ and $\mid\sim$ a (sound) inference relation or the semantic consequence relation.

Then the mapping $\mathbf{cl_K}^{\mid\sim} : \mathbf{IS_K}(\Sigma) \rightarrow \mathbf{IS_K}(\Sigma)$ is conservative, monotonic and idempotent.

Moreover, for any $\mathbf{M} \in \mathbf{IS_K}(\Sigma)$ and any model **N** of **K** that is a complete refinement of **M** it holds $\mathbf{cl_K}^{\mid\sim}(\mathbf{M}) \leq \mathbf{N}$. In particular this holds for the semantic closure mapping.

3.2 Object level reasoning traces and controlled inference functions

In Subsection 3.1 we have assumed that the deduction is exhaustive with respect to the specific set **K**; this is not a realistic assumption. In practice often only some of the inferences that are possible are applied, depending on additional control information.

However, the full deductive closure always gives an upper bound: if control is involved leading to non-exhaustive reasoning, the actual outcome is a model $\mathbf{M'}$ with

$$\mathbf{M} \leq \mathbf{M'} \leq dc_{\mathbf{K}}^{\vdash}(\mathbf{M}) \leq sc_{\mathbf{K}}(\mathbf{M})$$

In this paper we will assume that controlled inference is deterministic, depending on an assignment of values to some set of control parameters. In that case controlled inference can be described as follows.

Definition 3.5 (controlled inference function)

Suppose $\mathbf{K}$ is a set of formulae of signature Σ and $\vdash\!\sim$ is a (sound) inference relation or the (semantic) entailment relation. The mapping $\alpha\colon \mathbf{IS_K(\Sigma)} \rightarrow \mathbf{IS_K(\Sigma)}$ is called a *controlled inference function* for $\mathbf{K}$ based on $\vdash\!\sim$ if it is conservative and monotonic and for all $\mathbf{M} \in \mathbf{IS_K(\Sigma)}$ it holds

$$\alpha(\mathbf{M}) \leq cl_{\mathbf{K}}^{\vdash\sim}(\mathbf{M}).$$

Notice that we do not require that a controlled inference function is idempotent. If reasoning is not exhaustive idempotency is often lost. Controlled inference functions can be viewed as functions $\alpha_{\mathbf{K}}{}^{\mathbf{N}}$ where instead of a general entailment relation $\vdash\!\sim$ a variant is used that is parameterized by certain control information $\mathbf{N}$. Two examples of control parameters and the corresponding inference functions are:

- information about which atoms are the *goals* for the reasoning
In this case the control information $\mathbf{N}$ expresses that the conclusions should be restricted to what already is available and the set of atoms $\mathbf{G}$; i.e.,

$$\alpha_{\mathbf{K}}{}^{\mathbf{N}}(\mathbf{M})(\mathbf{a}) = \quad \mathbf{M(a)} \qquad\qquad \text{if } \mathbf{M(a)} \neq \mathbf{u}$$

$$cl_{\mathbf{K}}{}^{\vdash\sim}(\mathbf{M})|\mathbf{G(a)} \qquad \text{otherwise}$$

- information about the *selection of elements of the knowledge base* to be used
Here the control information $\mathbf{N}$ expresses that only formulae of a subset $\mathbf{K'}$ of the theory $\mathbf{K}$ can be used in the reasoning, i.e.,

$$\alpha_{\mathbf{K}}{}^{\mathbf{N}}(\mathbf{M}) = cl_{\mathbf{K'}}{}^{\vdash\sim}(\mathbf{M}) \leq cl_{\mathbf{K}}{}^{\vdash\sim}(\mathbf{M})$$

Notice that these examples of control apply not only to the case of an inference relation but also to the semantic consequence relation. In this sense control can be defined in a semantic (inference relation independent) manner.

In a meta-level architecture the control information $\mathbf{N}$ is determined by the meta-level reasoning. What is needed to formalize a meta-level architecture is a formalization of this control information on the right level of abstraction; i.e., in such a manner that it can be subject of a (meta-level) inference process. We will come back to this point in Section 4.

Definition 3.6 (object-reasoning trace)

Let $\langle \Sigma_0, \mathbf{OT}, \vdash \rangle$ be a static view on the object-level reasoning component, where $\vdash$ is a sound inference relation. A partial temporal model $(\mathbf{M_t})_{t \in N}$ is called an *(object-reasoning) trace* for $\langle \Sigma_0, \mathbf{OT}, \vdash \rangle$ if for all $\mathbf{s}, \mathbf{t} \in \mathbf{N}$ with $\mathbf{s} < \mathbf{t}$ it holds

$$\mathbf{M_s} \leq \mathbf{M_t} \leq \mathbf{dc_{OT}}^{\vdash}(\mathbf{M_s}).$$

Theorem 3.7 (approximation of an intended model: soundness)

Let $\langle \Sigma_0, \mathbf{OT}, \vdash \rangle$ be a static view on the object-level reasoning component and $\mathbf{N}$ a model of $\mathbf{OT}$ (the intended model).

a) If $(\mathbf{M_t})_{t \in \mathbb{N}}$ is a trace for $\langle \Sigma_0, \mathbf{OT}, \vdash \rangle$ with root $\mathbf{r}$ and $\mathbf{M_r} \leq \mathbf{N}$ then for all $\mathbf{t} \in \mathbf{N}$ it holds $\mathbf{M_t} \leq \mathbf{N}$, i.e.:

$$\mathbf{M_r} \leq \ ... \leq \mathbf{M_t} \leq ... \leq \mathbf{N}$$

b) Let for any $\mathbf{t} \in \mathbf{N}$ a controlled inference function $\alpha_t : \mathbf{IS_{OT}}(\Sigma_0) \to \mathbf{IS_{OT}}(\Sigma_0)$ for $\mathbf{OT}$ be given.

Then for any starting point $\mathbf{M_0} \in \mathbf{IS_K}(\Sigma)$ a trace $(\mathbf{M_t})_{t \in N}$ for $\langle \Sigma_0, \mathbf{OT}, \vdash \rangle$ can be defined by:

$$\mathbf{M_{t+1}} = \alpha_t(\mathbf{M_t}) \qquad \text{for all } \ \mathbf{t} \in \mathbf{N}$$

Given the formal framework as set up here, the proof of this theorem is not difficult. The above results show a direct connection between the semantics on the basis of partial models (as used here) and the classical Tarski semantics. In our terms this connection can be stated as follows. Reasoning of the object-level component is always on one specific, intended (complete) model that is a (Tarski) model of the knowledge base. An information state is a partial description of this intended model: a partial model with the intended model as one of its complete refinements. During reasoning this partial description is (step-wise) refined, but (in sound reasoning processes) always remains within the intended model. In our approach reasoning can be viewed as constructing a partial model, approximating the intended complete model better and better by refinement steps. This even holds if additional observations are allowed, based on the intended model (this point is left out of the current paper). Since at any moment in time the intended complete model is not known, in principle we have to take into account all complete refinements of the current information state that are models of the knowledge base. Thus the approach discussed here relates *static semantics* and *dynamic semantics* to each other in one formal framework.

3.3 Control information and dynamic view on object-level reasoning

In this section we will introduce a formalization of control aspects of the object-level reasoning. The intended model of the object-level component is (a formal representation of) a specific world situation. As the meta-level component reasons about the reasoning process of the object-level component, the intended model of this is a formal description of (relevant aspects of) the inference process of the object-level component. Considered from the viewpoint of the meta-level component, the object-level component can be considered as some exotic world situation with as a crucial

characteristic that it is dynamic: each time the meta-level component starts a new reasoning session, its associated world situation may have changed. Note that we assume that object-level and meta-level reasoning processes are alternating: during the meta-level reasoning the object-level component is not reasoning, so changes of the object-level state occur only between the reasoning sessions of the meta-level component.

This observation leads us to introduce a *control signature* that defines at an abstract level a number of descriptors that can be used to characterize the control and process states of the object-level reasoning: a lexicon in terms of which all relevant control information can be expressed. A truth assignment to the ground atoms of such a meta-signature is called a *control-information state*. Such a control-information state can serve as a (partial) model for the meta-level component. The question of what are the semantics of the meta-level component is equivalent to the question of what is described by the control-information state related to an object-level component. We illustrate this idea by some examples (for a more specific example, see Section 6):

- the fact that the object-level statement **h** is (currently) considered a *goal* for the reasoning process; e.g., expressed by the (ground) control-atom **goal(h)** where **h** is the name of an atom in the object-level language;
- a *selection* or *priority* of object-level knowledge elements to be used; e.g., expressed by the (ground) control-atom **rule_priority(r)**, where **r** is the name of a rule in the object-level knowledge base, or **goal_priority(h, 0.9)**, with **h** as above;
- the degree of *exhaustiveness* of the reasoning; e.g., expressed by **exhaustiveness(any)**, meaning that it is enough to determine only one of the current goals (the one with highest possible priority).

A control-information state formalizes at a high level of abstraction the parameter **N** in a controlled inference function as introduced earlier in Section 3. We assume that the control-information state specifies all information relevant to the control of the (future) reasoning behaviour; i.e., the object-information state and the control-information state together determine in a deterministic manner the behaviour of the object-level reasoning component during its next activation. Of course it depends on the specific inference procedure that is used which control aspects can be influenced and which aspects cannot.

In principle for execution we would expect that all atoms of the control signature have a truth value assigned to it (i.e. the control-information state is a complete model). However, we allow partial control-information states as well.

Definition 3.8 (dynamic view on the object-level component)
A *dynamic view* related to the static view $\langle \Sigma_o, \mathbf{OT}, \vdash \rangle$ on an object-level component is a tuple $\langle \Sigma_c, \mu_{\mathbf{OT}}^{\vdash}, \upsilon_{\mathbf{OT}}^{\vdash} \rangle$ with Σ_c a signature called *control signature* and $\mu_{\mathbf{OT}}^{\vdash}$, $\upsilon_{\mathbf{OT}}^{\vdash}$ mappings

$$\mu_{\mathbf{OT}}^{\vdash} \quad : \mathbf{IS}(\Sigma_o) \times \mathbf{IS}(\Sigma_c) \rightarrow \mathbf{IS}(\Sigma_o)$$
$$\upsilon_{\mathbf{OT}}^{\vdash} \quad : \mathbf{IS}(\Sigma_o) \times \mathbf{IS}(\Sigma_c) \rightarrow \mathbf{IS}(\Sigma_c)$$

We call $\mu_{OT}^{\vdash}$ the *(controlled) inference function* for the object-level, and $\upsilon_{OT}^{\vdash}$ the *process state update function*. For any $N \in IS(\Sigma_c)$ the mapping

$$\mu_{OT}^N : IS(\Sigma_o) \rightarrow IS(\Sigma_o)$$

is defined by

$$\mu_{OT}^N : M \mapsto \mu_{OT}^{\vdash}(M, N)$$

We assume that for any $N \in IS(\Sigma_c)$ this μ_{OT}^N is a controlled inference function, i.e., it is conservative and monotonic and satisfies

$$\mu_{OT}^N(M) \leq dc_{OT}^{\vdash}(M) \text{ for all } M \in IS(\Sigma_o).$$

When no confusion is expected, we will leave out the subscript and superscript of $\mu_{OT}^{\vdash}$ and $\upsilon_{OT}^{\vdash}$ and write shortly μ and υ.

In a control signature sometimes reference will be made to (names of) elements of the language based on the object-level signature. On the other hand, also control-atoms are possible that do not refer to specific object-level language elements (e.g., exhaustiveness). We do not prescribe in a generic manner if and how reference is made to object-level language elements. In examples this will always be determined in a more specific manner.

The process state update function expresses what the process brings about with respect to the process state descriptors. Examples: an object-atom was unknown, but becomes known during the reasoning; an object-atom that was a goal has failed to be found.

The functions μ and υ for partial control-information states can be defined from the values of the functions for complete control-information states as follows:

$$\mu(M, N) = gci\{\mu(M, N') \mid N \leq N' \& N' \text{ complete}\}$$
$$\upsilon(M, N) = gci\{\upsilon(M, N') \mid N \leq N' \& N' \text{ complete}\}$$

where the greatest common information state $gci(S)$ of a set S of information states is defined by

$$gci(S)(a) = \begin{cases} 1 & \text{if for all } M \in S \text{ it holds } M(a) = 1 \\ 0 & \text{if for all } M \in S \text{ it holds } M(a) = 0 \\ u & \text{otherwise} \end{cases}$$

A (combined) information state is a pair $\langle M, N \rangle$ where M is an object-information state and N a control-information state. A (combined) trace is a sequence of (combined) information states. An object-level *execution step* on the basis of a combined information state $\langle M_t, N_t \rangle$ provides the *successor* information state defined by:

$$\langle M_{t+1}, N_{t+1} \rangle = \langle \mu(M_t, N_t), \upsilon(M_t, N_t) \rangle$$

A combined reasoning trace can be obtained by alternating object-level execution steps and interaction steps between the two levels to obtain new control-information states **N**. We will work this out in more detail in Section 5.

4 Temporal interpretation of the meta-level reasoning

Locally, at each of the two reasoning levels, the system behaves conservative and monotonic, but the whole cycle implies non-conservative changes of information states: the actions induced by the upward and downward reflections are not conservative. To describe this we can label the information states with an explicit time parameter (e.g., expressed by natural numbers). The non-conservatism can be covered by our declarative formal model, assuming each new (object-meta) cycle is labeled with the next (successor) time label. This approach implies that the meta-level reasoning component has semantics that relates states of the object-level reasoning component at time **t** to states of this component at time **t + 1**. In this manner, statements like

  "*if the atom* **a** *is unknown, then the atom* **b** *is proposed as a goal*"

after downward reflection can be interpreted in a temporal manner:

  "*If at time* **t** *the atom* **a** *is unknown*   *(in the object-level reasoning component)*
   then at time **t + 1** *the atom* **b** *is a goal* "   *(for the object-level reasoning process)*

Assuming that the meta-level's proposals are always accepted (this assumption is sometimes called causal connection), downward reflection is just a shift in time, replacing the goals at the object-level by new goals (the ones proposed by the meta-level). This can be expressed as follows

  $\neg$ **known(a)** $\rightarrow$ **proposed_goal(b)**   (meta-knowledge)

  C(proposed_goal(b)) $\rightarrow$ **X(goal(b))**   (downward reflection)

where **C** means "holds in the current state" and **X** "holds in the next state".

  Within the *meta-information* involved in the meta-reasoning we distinguish two special types: a) information on relevant aspects of the *current* (control-)state of the object-level reasoning process (possibly also including facts inherited from the past), and b) information on proposals for control parameters that are meant to guide the object-level reasoning process in the near *future*. Therefore we assume that in the meta-signature a copy of the control signature of the object-level component is included as a subsignature that refers to the current state. Moreover, we assume that a second copy of this control signature is included referring to the proposed truth values for the next state of the object-level reasoning process. For example, if **goal(h)** is an atom of the control signature, then there are copies **current_goal(h)** and

proposed_goal(h) in the set of atoms for the meta-signature. A syntactic function transforming a meta-atom into a current variant and a proposed variant of it can simply be defined by two (injective) mappings **c** and **p** of predicates, leaving the arguments the same e.g.,

$$\textbf{c(goal) = current\_goal, p(goal) = proposed\_goal}.$$

We assume that the reasoning of the meta-level itself has no sophisticated control: for simplicity we assume that it concerns taking deductive closures with respect to the inference relation used at the meta-level. Under this assumption a dynamic view on the meta-level component is completely determined by a static view.

Definition 4.1 (static and dynamic view on the meta-level component)

a) The signature Σ_m is called a *meta-signature* related to Σ_c if there are two injective mappings $\mathbf{c} : \Sigma_c \to \Sigma_m$ and $\mathbf{p} : \Sigma_c \to \Sigma_m$. In this case the subsignatures $\mathbf{c}(\Sigma_c)$ and $\mathbf{p}(\Sigma_c)$ are denoted by $\Sigma_m{}^c$ and $\Sigma_m{}^p$; they are referring to the *current state control-information* of the object-level and the *proposed state control-information* for the object-level.

b) The *static view on the meta-level component* is a tuple

$$\langle\, \Sigma_m, \textbf{MT}, \vdash_m \,\rangle$$

with

Σ_m a signature, called the meta-signature related to Σ_c

MT   a set of propositional ground formulae expressed in terms of the meta-signature

$\vdash_m$ a classical inference relation (assumed sound but not necessarily complete)

c) The *inference function of the meta-level* $\mu_{MT}{}^{\vdash}{}_m$ (or shortly μ^*)

$$\mu_{MT}{}^{\vdash}{}_m\colon \textbf{IS}_{MT}(\Sigma_m{}^c) \to \textbf{IS}_{MT}(\Sigma_m)$$

is defined by the exhaustive inference function based on $\vdash$; i.e., by the transition function

$$\mu_{MT}{}^{\vdash}{}_m : \textbf{N} \mapsto \textbf{dc}_{MT}{}^{\vdash}{}_m(\textbf{N})$$

This function μ^* defines the *dynamic view on the meta-level component*, related to the static view

$$\langle\, \Sigma_m, \textbf{MT}, \vdash_m \,\rangle.$$

Note that we essentially use propositional logic to describe the meta-language; if needed a propositional signature can be defined based on the set of all ground atoms expressible in a given (many-sorted) predicate logic signature. In fact it does not matter how language elements at the meta-level are denoted, but how their semantics is defined (in terms of the controlled inference function).

5 Temporal models of overall reasoning patterns

After having introduced the required concepts in the previous sections, in this section it will turn out to be easy to compose them to semantics for the dynamics of a meta-level architecture. The information states of the meta-level component of the reasoning system will have a direct impact on the control-information state of the object-component, and vice versa. These connections will be defined formally in this section. Notice that in our approach the object-level component and the meta-level component do not reason at the same time, but are alternating.
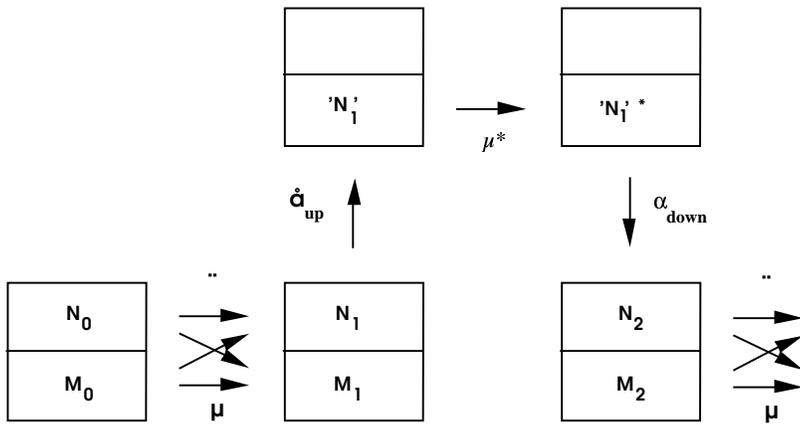

Fig 2 Transitions of information states in a meta-level architecture

In fact the following four types of actions take place (see Fig. 2). For a formal description, see Definitions 5.1 and 5.2 below.

  - *object-level reasoning*
  The reasoning of the object-level component can be described by the functions μ and υ as defined in Definition 3.8.

  - *upward reflection*
  Information from the control-information state of the object-level component is transformed (by a transformation function α_{up} defined in Definition 5.1 below) to the next information state of the meta-level component. This will provide input information for the subsequent reasoning of the meta-level component (see Definition 4.1).

- *meta-level reasoning*

The reasoning of the meta-level component can be described by the inference function μ^* as defined in Definition 4.1.

- *downward reflection*

Information from the of the meta-level component is transformed (by the mapping α_{down}; see Definition 5.1 below) to the next control-information state to be used in the control of the object-level component. This will affect the reasoning behaviour during the subsequent object-level reasoning.

Definition 5.1 (meta-level architecture for dynamic control)

a) A *meta-level architecture for dynamic control* is described by a tuple

$$\mathbf{MLC} = \langle\, \langle\, \Sigma_o, \mathbf{OT}, \vdash_o \,\rangle; \langle\, \Sigma_c, \mu, \upsilon \,\rangle; \langle\, \Sigma_m, \mathbf{MT}, \vdash_m \rangle \; ; \langle\, \mathbf{c}, \mathbf{p} \,\rangle\,\rangle$$

where

$$\langle\, \Sigma_o, \mathbf{OT}, \vdash_o \,\rangle$$
$$\langle\, \Sigma_c, \mu, \upsilon \,\rangle$$

are a static and a dynamic view on the object-level component,

$$\langle\, \Sigma_m, \mathbf{MT}, \vdash_m \,\rangle$$

is a static view on the meta-level component, where Σ_m is related to the control signature Σ_c by the injective functions $\mathbf{c} : \Sigma_c \to \Sigma_m$ and $\mathbf{p} : \Sigma_c \to \Sigma_m$ and $\vdash_m$ is an inference relation. Moreover, $\mathbf{MT}$ is a meta-knowledge base satisfying

$$\mathbf{IS_{MT}}(\Sigma_m{}^c) = \mathbf{IS}(\Sigma_m{}^c)$$

i.e., no information state in $\mathbf{IS}(\Sigma_m{}^c)$ is inconsistent with $\mathbf{MT}$.

Based on $\mathbf{MLC}$ we can define the function μ^* according to Definition 4.1.

b) Let $\mathbf{MLC}$ be as in a). The *upward reflection function* is the mapping

$$\alpha_{up} \colon \mathbf{IS}(\Sigma_c) \to \mathbf{IS}(\Sigma_m)$$

defined for $\mathbf{N} \in \mathbf{IS}(\Sigma_c)$ and $\mathbf{b} \in \mathbf{At}(\Sigma_m)$ by

$$\alpha_{up}(\mathbf{N})(\mathbf{b}) \quad = \qquad \mathbf{N(a)} \qquad \text{if } \mathbf{b} = \mathbf{c(a)} \text{ for some } \mathbf{a} \in \mathbf{At}(\Sigma_c)$$
$$\mathbf{u} \qquad \text{otherwise}$$

The *(left) inverse upward reflection function* β is the mapping

$$\beta \colon \mathbf{IS}(\Sigma_m) \to \mathbf{IS}(\Sigma_c)$$

defined for $\mathbf{N} \in \mathbf{IS}(\Sigma_m)$ and $\mathbf{a} \in \mathbf{At}(\Sigma_c)$ by

$$\beta(\mathbf{N})(\mathbf{a}) \qquad = \qquad \mathbf{N(c(a))}$$

The *downward reflection function* is the mapping

$$\alpha_{down} \colon \mathbf{IS}(\Sigma_m) \to \mathbf{IS}(\Sigma_c)$$

defined for $\mathbf{N} \in \mathbf{IS}(\Sigma_m)$ and $\mathbf{a} \in \mathbf{At}(\Sigma_c)$ by

$$\alpha_{down}(\mathbf{N})(\mathbf{a}) = \qquad \mathbf{1} \qquad\quad \text{if } \mathbf{N(p(a))} = \mathbf{1}$$
$$\mathbf{0} \qquad\quad \text{otherwise}$$

The *time shift function* is the mapping

$$\sigma \colon \mathbf{IS}(\Sigma_m) \to \mathbf{IS}(\Sigma_m)$$

defined by

$$\sigma(N)(b) \;\; = \;\; N(p(a)) \qquad \text{if} \;\; b = c(a) \;\; \text{for some} \;\; a \in At(\Sigma_c)$$
$$\qquad\qquad\qquad\qquad\qquad \text{and} \;\; N(P(a)) \neq u$$
$$\qquad\qquad\quad\; 0 \qquad\qquad\; \text{if} \;\; b = c(a) \;\; \text{for some} \;\; a \in At(\Sigma_c)$$
$$\qquad\qquad\qquad\qquad\qquad \text{and} \;\; N(P(a)) = u$$
$$\qquad\qquad\quad\; u \qquad\qquad\; \text{otherwise}$$

Reasoning activities are modifying object-information states in a conservative manner (making refinements). Notice, however, that execution of upward and downward reflection may induce non-conservative changes. Notice that we force the new control state resulting from downward reflection to be two-valued. This is to avoid nondeterministic phenomena and to allow that the meta-level only provides the relevant (partial) information on control. In the rest of the paper **MLC** will denote a tuple as defined in Definition 5.1.

The following relations hold for the functions defined above:
$$\beta\alpha_{up} \; = id, \; \alpha_{down} = \beta\sigma, \; \sigma = \alpha_{up}\alpha_{down}.$$

Definition 5.2 (overall semantics based on traces)
a) An *overall trace* for the meta-level architecture **MLC** is a labeled linear partial temporal model
$$(M_t \oplus N_t)_{t \in N}$$
(also denoted by **M** ⊕ **N**) over $IS(\Sigma_o \oplus \Sigma_m)$ with set of labels $L = \{re, sh\}$ (**re** for a reasoning step, **sh** for a time shift step) satisfying the following conditions for all $s, t \in N$:
 (i) If $s <_{re} t$
$$M_t \quad = \quad \mu(M_s, \beta(N_s))$$
$$N_t \quad = \quad \mu^*(\alpha_{up}(\upsilon(M_s, \beta(N_s))))$$
(ii) If $s <_{sh} t$
$$M_t \quad = \quad M_s$$
$$N_t \quad = \quad \mu^*(\sigma(N_s))$$
b) The *(intended) semantics* of the meta-level architecture **MLC** is the set of traces as defined in a), denoted by **Traces(MLC)**.
c) A trace is called *alternating* if for all $r, s, t \in N$ and $i, j \in L$ with $r <_i s <_j t$ it holds $i \neq j$.

Although usually we are most interested in alternating traces, there may be cases were we are interested in other traces as well: e.g. if we allow multiple activations of the object-level without intervenience of the meta-level.

Temporal models can be defined using our framework by traces of information states. These traces are constructed during reasoning. At each moment of time only a partial (in time) fragment of such a trace-model has been constructed. The set of completed traces can be viewed as the set of intended overall models of the meta-level architecture. The meta-level architecture as a whole approximates an intended model

in a conservative manner by subsequently adding elements to the trace according to time steps. This view will be made more precise in the following theorem.

Theorem 5.3 (approximation of a trace)
Let **MLC** be a meta-level architecture for dynamic control.
a) The set of alternating traces of **MLC** is parameterized by the initial states, together with the label (from **{re, sh}**) of the initial transition.
b) Let $\mathbf{M} \oplus \mathbf{N}$ be a trace for **MLC**. Define for any time point **t**

$$\mathbf{M}^{(t)}_s(\mathbf{a}) \quad = \qquad \mathbf{M}_s(\mathbf{a}) \qquad \text{if } \mathbf{s} \ll \mathbf{t} \text{ or } \mathbf{s} = \mathbf{t}$$
$$\mathbf{u} \qquad\qquad \text{otherwise}$$
$$\mathbf{N}^{(t)}_s(\mathbf{a}) \quad = \qquad \mathbf{N}_s(\mathbf{a}) \qquad \text{if } \mathbf{s} \ll \mathbf{t} \text{ or } \mathbf{s} = \mathbf{t}$$
$$\mathbf{u} \qquad\qquad \text{otherwise}$$

Then for all time points **t** it holds

$$\mathbf{M}^{(0)} \oplus \mathbf{N}^{(0)} \leq \dots \leq \mathbf{M}^{(t)} \oplus \mathbf{N}^{(t)} \leq \mathbf{M}^{(t+1)} \oplus \mathbf{N}^{(t+1)} \leq \dots \leq \mathbf{M} \oplus \mathbf{N}$$

Notice that in this section we do not (yet) add temporal elements to the languages of the reasoning components themselves, but we attribute temporal semantics to the whole system by interpreting the object reasoning process and the downward reflection in a temporal manner. This means that within each of the components (locally) we retain our original (non-temporal) semantics. The temporal semantics only serves as a foundation for the composition principle to define an overall semantics composed from the local semantics of each of the components at the two levels.

6 An example reasoning pattern

To illustrate the concepts introduced here we will give a trace of a meta-level architecture for reasoning with dynamic hypotheses that are used as goals. The meta-level reasoning performs hypothesis selection whereas the object-level reasoning performs testing of hypotheses by trying to derive them from observation information (in a goal-directed manner). Control is needed to direct the object-level reasoning to the goal that is to be posed: the selected hypothesis. The meta-level contains declarative knowledge on which hypothesis to select under which circumstance (state of the object-level reasoning process). The downward reflection transforms this information about the selected hypothesis to control-information in the form of a goal set for the object-level reasoning; this enables the system to effectuate control. The upward reflection provides the information for the meta-level component on the current state of what is already known and what is not yet known in the object-level reasoning process. The knowledge in this example system is not realistic, but it enables one to get an impression of the reasoning pattern.

A. *Static view on the object-level reasoning component*

Object-signature (propositional):

$$\Sigma_o = \langle\, s_1, s_2, s_3, h_1, h_2\, \rangle$$

Object theory (knowledge base of the object-level component) **OT**:

$$
\begin{aligned}
s_2 \wedge s_3 &\;\rightarrow\; h_1 \\
\neg\, s_3 \wedge s_1 &\;\rightarrow\; h_2 \\
\neg\, s_3 &\;\rightarrow\; \neg\, h_1
\end{aligned}
$$

Inference relation: $\vdash_{\mathbf{ch}}$ (chaining)

B. *Dynamic view on the object-level reasoning component*

This reasoning component is used in a goal-directed fashion with *chaining* as *inference relation*. We will not involve the possibility to acquire additional information from the outside of the system.

Control-signature
$$\langle\, \textbf{true\_s}_1, \textbf{false\_s}_1, \textbf{true\_s}_2, \textbf{false\_s}_2, \textbf{true\_s}_3, \textbf{false\_s}_3,$$
$$\textbf{known\_h}_1, \textbf{known\_h}_2, \textbf{goal\_h}_1, \textbf{goal\_h}_2\, \rangle$$

Inference function
The dependency of the *inference function* μ on the control-information state is concentrated in information expressed by goal-statements **goal_h$_i$** (meaning that **h$_i$** is a goal for the object-level reasoning process). As a formal definition we can take

$$\mu(\textbf{M}, \textbf{N})(\textbf{a}) = \quad \textbf{dc}_{\textbf{OT}}{}^{\vdash}{}_{\textbf{ch}}(\textbf{M})(\textbf{a}) \quad \text{if} \quad \textbf{a} = \textbf{h}_i \text{ and } \textbf{N}(\textbf{goal\_h}_i) = \textbf{1}$$
$$\textbf{M(a)} \qquad\qquad \text{otherwise}$$

Process state update function
The *process state update function* υ is defined as follows:
  - The statement **known_h$_i$** gets truth value **1** in the control-information state if in the object-information state **h$_i$** has truth value **1** or **0**; it gets truth value **0** otherwise (i.e., if **h$_i$** has truth value **u** in the object-information state).
  - The statement **true_s$_i$** has truth value **1** in the control-information state if in the object-information state **s$_i$** has truth value **1**; it gets truth value **0** otherwise (i.e., if **s$_i$** has truth value **u** or **0** in the object-information state).

- The statement **false_s$_i$** has truth value **1** in the control-information state if in the object-information state **s$_i$** has truth value **0**; it gets truth value **0** otherwise (i.e., if **s$_i$** has truth value **u** or **1** in the object-information state).
- The other truth values remain unchanged.

C. *Meta-level reasoning component*

The *meta-signature* is taken as the disjoint union of two copies of the control signature above: **c_at** (currently **at**), and **p_at** (proposed **at**), where **at** is an atom of the control signature.
Knowledge base of the meta-level component (**MT**):

$$\textbf{c\_true\_s}_2 \ \land \ \neg \ \textbf{c\_known\_h}_1 \ \rightarrow \ \textbf{p\_goal\_h}_1$$
$$\textbf{c\_false\_s}_3 \ \land \ \neg \ \textbf{c\_known\_h}_2 \ \rightarrow \ \textbf{p\_goal\_h}_2$$

The meta-level wil use *chaining* as its *inference relation*.
The inference function $\mu*$ is the deductive closure function under **MT**.

Trace of an example session

In Fig. 3 a session with initial state $\langle$ **s$_1$, s$_2$, s$_3$** $\rangle$ **:** $\langle$ **1, 1, 0** $\rangle$ is described. Here for convenience partial models are denoted by the list of atomic statements and negations of atomic statements that are true. For the (combined) information states (named **p$_i$**) of the object-level component both the object-information states and the control-information states are depicted (separated by a colon **;**). For the meta-level component only the object-information states are depicted (named **t$_i$**). For shortness only some relevant (literal) facts are written in the information states.

object-level component   *meta-level component*

p$_0$: [s$_1$, s$_2$, ¬ s$_3$]; [¬ known_h$_1$, ¬ known_h$_2$]
p$_1$: [s$_1$, s$_2$, ¬ s$_3$]; [true_s$_2$, ¬ known_h$_1$, ¬ known_h$_2$]

                  t$_0$: [c_true_s$_2$, ¬ c_known_h$_1$, ¬ c_known_h$_2$]
                  t$_1$: [c_true_s$_2$, ¬ c_known_h$_1$, ¬ c_known_h$_2$, p_goal_h$_1$]

p$_2$: [s$_1$, s$_2$, ¬ s$_3$]; [true_s$_2$, ¬ known_h$_1$, ¬ known_h$_2$, goal_h$_1$]
p$_3$: [s$_1$, s$_2$, ¬ s$_3$, ¬ h$_1$]; [true_s$_2$, false_s$_3$, known_h$_1$, ¬ known_h$_2$]

                  t$_2$: [c_true_s$_2$, c_false_s$_3$, c_known_h$_1$, ¬ c_known_h$_2$]
                  t$_3$: [c_true_s$_2$, c_false_s$_3$, c_known_h$_1$, ¬ c_known_h$_2$, p_goal_h$_2$]

p$_4$: [s$_1$, s$_2$, ¬ s$_3$, ¬ h$_1$]; [true_s$_2$, false_s$_3$, known_h$_1$, ¬ known_h$_2$, goal_h$_2$]
p$_5$: [s$_1$, s$_2$, ¬ s$_3$, ¬ h$_1$, h$_2$]; [true_s$_1$, true_s$_2$, false_s$_3$, known_h$_1$, known_h$_2$]

Fig 3 Trace of an example session

This example shows that it does not matter how language elements at the meta-level are denoted, but how their semantics is defined. Using a propositional language at the meta-level is possible, but the more concise syntactical notation of predicate logic has practical advantages. Therefore, often a predicate logic language is used at the meta-level. For the semantics of the whole reasoning pattern this makes no essential difference.

7 Conclusions

The semantic framework as discussed provides integration of static and dynamic aspects in two different forms. On the one hand we connect the partial semantics as used to describe information states to the standard Tarski semantics: a partial model corresponds to the set of all of its complete refinements. On the other hand the integration between static and dynamic aspects takes place by introducing the notion of an explicit (declarative) control-information state in the object-level component.

Our logical framework has been partly inspired by Weyhrauch's view on the role of partial models (or simulation structures) in meta-level architectures ([37], [19]); see also [32]. What is different in our case is that the partial models may be dynamic. Furthermore, similarities can be found to the approach called dynamic interpretation of natural language (e.g., see [17], [20], [23]). In this approach the dynamic interpretation of a sentence in natural language is defined as an operator that transforms the current information state into a new one where the content of the sentence is included.

With respect to dynamics the type of meta-level architecture covered here is less restricted than sometimes studied in logical approaches, where meta-level predicates are meant to express only static properties of the object-level, e.g., provability, et cetera. We believe that the semantic model as discussed here can help to bridge the gap between the (more restricted) logic-based approaches and (less restricted) procedural approaches to meta-level architectures.

It is not difficult to use our framework to model meta-level reasoning that looks ahead more than one step. One can transfer a part of the information at the meta-level over the time shift and thus connect the reasoning in different activations of the meta-level. The results presented here can also be extended easily to the case of higher meta-levels where also the control of the meta-level is guided in a dynamic manner (in this case a refinement of the time scale can be made).

The type of meta-level architecture addressed by the semantic framework has been implemented and applied in a number of practical applications, often in projects in cooperation with companies (e.g., [6], [7], [8], [9], [18]). The type of meta-level architecture as discussed can be designed and formally specified using our compositional design method DESIRE (DEsign and Specification of Interacting REasoning components; see [5], [26]). By means of DESIRE complex reasoning systems or agents can be designed and specified according to what we call a

compositional architecture: an architecture composed from a number of formally specified reasoning components using formal composition principles (see [5]). In DESIRE various types of reasoning components are covered; e.g., goals can be used to guide the reasoning, and various measures of exhaustiveness can be used for the control of reasoning (for more details, see [5]).

In our current practical applications (using DESIRE) of the framework as described the control-information states are two-valued, i.e., no **u**'s occur as truth values. In other words: at each moment all state descriptors defined by the control signature have a determined truth value. This corresponds to the intuition that, since the meta-level reasoning is about the states of the object-level reasoning process, input information about this can be acquired from the system itself: there is essentially no incompleteness of incoming information. On the other hand the two-valuedness of the control-information states is related to the fact that we require that the control of the object-level reasoning is completely determined by the truth values of the control-atoms, and vice versa. Therefore, if we require complete deterministic specification of the behaviour of the reasoning system, all control-atoms should have determined truth values (and vice versa). If we would allow non-deterministic control (e.g., by only specifying some, but not all aspects of the control), the control-information states may be viewed as essentially three-valued.

Since for the notion of a compositional architecture and the design method DESIRE, an essential use is made of the notion of a meta-level architecture, the (formal) semantics of the static and dynamic semantics of DESIRE depend on these semantics of meta-level architectures. In the literature not much work is reported on such foundations. As this paper contributes formal semantics of meta-level architectures, this can be used for a semantics of DESIRE.

Meta-level architectures have been exploited to model nonmonotonic reasoning (see [1], [30], [31]). The type of architecture there is based on temporal epistemic reflection (i.e., dynamic addition and retraction of assumptions by explicit meta-reasoning), and not on meta-level control of the object-level reasoning (which takes place as a non-controlled deductive closure determination; therefore it differs from the type of meta-level architecture considered here. Formal analysis of semantics of this type of architecture was addressed in [21]. An interesting combination would be if both meta-reasoning on assumptions and meta-reasoning on control of the object reasoning would be covered in one (combined) architecture; this has not been addressed yet.

Disjoint from the area of meta-level architectures, a temporal perspective on the semantics of reasoning processes has been very fruitful to obtain semantics, temporal specification languages, and simulation environments for of nonmonotonic reasoning processes; cf. [13], [14], [16]. A specific type of nonmonotonic reasoning is based on default logic. In [12], [15], [22] different aspects of semantics and specification of default reasoning processes have been analysed in more depth.

Acknowledgements

References

1. Allis V.E., Tan Y.H., Treur J., Meta-level Selection Techniques for the Control of Default Reasoning. *Future Generation Computer Systems*, vol. 12, Special double issue (2-3): Reflection and Meta-level AI Architectures, (R. Lopez de Mantaras, ed.), 1996, pp. 189-201.

2. J.F.A.K. van Benthem, The logic of time: a model-theoretic investigation into the varieties of temporal ontology and temporal discource, Reidel, Dordrecht, 1983.

3. S. Blamey, Partial Logic, in: D. Gabbay and F. Guenthner (eds.), Handbook of Philosophical Logic, Vol. III, 1-70, Reidel, Dordrecht, 1986.

4. K. Bowen and R. Kowalski, Amalgamating language and meta-language in logic programming. In: K. Clark, S. Tarnlund (eds.), Logic programming. Academic Press, 1982.

5. Brazier, F.M.T., Jonker, C.M., and Treur, J., Principles of Compositional Multi-agent System Development. In: J. Cuena (ed.), *Proceedings of the 15th IFIP World Computer Congress, WCC'98, Conference on Information Technology and Knowledge Systems, IT&KNOWS'98,* 1998, pp. 347-360. To be published by IOS Press. Brazier,

6. F. M. T., Jonker, C. M., Treur, J., and Wijngaards, N.J.E, (2000), On the Use of Shared Task Models in Knowledge Acquisition, Strategic User Interaction and Clarification Agents. *International Journal of Human-Computer Studies,* vol. 52, 2000, pp. 77-110.

7. Brazier, F.M.T., Langen, P.H.G. van, and Treur, J., Strategic Knowledge in Design: a Compositional Approach. *Knowledge-based Systems,* vol. 11, 1998 (Special Issue on Strategic Knowledge and Concept Formation, K. Hori, ed.), pp. 405-416.

8. Brazier, F.M.T., and Treur J., Compositional Modelling of Reflective Agents. *International Journal of Human-Computer Studies,* vol. 50, 1999, pp. 407-431.

9. H.A. Brumsen, J.H.M. Pannekeet and J. Treur, A compositional knowledge-based architecture modelling process aspects of design tasks, Proc. 12th Int. Conf. on AI, Expert systems and Natural Language, Avignon'92 (Vol. 1), 1992, pp. 283-294.

10. W.J. Clancey and C. Bock, Representing control knowledge as abstract tasks and metarules, in: Bolc, Coombs (eds.), Expert System Applications, 1988.

11. R. Davis, Metarules: reasoning about control, Artificial Intelligence 15 (1980), pp. 179-222.

12. Engelfriet, J., Marek, V.W., Treur, J., and Truszczinski, M., Default Logic and Specification of Nonmonotonic Reasoning. *Journal of Experimental and Theoretical AI.* To appear.

13. Engelfriet, J., and Treur, J., Temporal Theories of Reasoning. *Journal of Applied Non-Classical Logics,* 5, 1995, pp. 239-261.

14. Engelfriet J., Treur J. Executable Temporal Logic for Nonmonotonic Reasoning. *Journal of Symbolic Computation*, vol. 22, 1996, pp. 615-625.

15. Engelfriet J. and Treur, J., An Interpretation of Default Logic in Minimal Temporal Epistemic Logic. *Journal of Logic, Language and Information*, vol. 7, 1998, pp. 369-388.

16. Engelfriet J., and Treur J. Specification of Nonmonotonic Reasoning. *Journal of Applied Non-Classical Logics*, vol. 10, 2000, pp. 7-27

17. T. Fernando, Transition systems and dynamic semantics, Proc. JELIA'92 Workshop on Logic and AI, Berlin, 1992.

18. P.A. Geelen and W. Kowalczyk, A knowledge-based system for the routing of international blank payment orders, Proc. 12th Int. Conf. on AI, Expert systems and Natural Language, Avignon-92 (Vol. 2), 1992, pp. 669-677.

19. E. Giunchiglia, P. Traverso and F. Giunchiglia, Multi-context Systems as a Specification framework for Complex Reasoning Systems, In: [36], 1993, pp. 45-72.

20. R. Goldblatt, Logics of Time and Computation. CSLI Lecture Notes, Vol. 7. 1987, Center for the Study of Language and Information.

21. Hoek, W. van der, Meyer, J.-J.Ch., and Treur, J., Formal semantics of temporal epistemic reflection. In: L. Fribourg and F. Turini (ed.), Logic Program Synthesis and Transformation-Meta-Programming in Logic, Proc. Fourth Int. Workshop on Meta-programming in Logic, META'94, Lecture Notes in Computer Science, vol. 883, Springer Verlag, 1994, pp. 332-352.

22. Hoek, W. van der, Meyer, J.J. Ch., Treur, J., Temporal Epistemic Default Logic. Journal of Logic, Language and Information , vol. 7, 1998, pp. 341-367.

23. J.A.W. Kamp, A theory of truth and semantic representation, In: Formal methods in the study of language. Mathematical Centre Tracts 135, Amsterdam, 1981.

24. P.H.G. van Langen and J. Treur, Representing world situations and information states by many-sorted partial models, Report PE8904, University of Amsterdam, Department of Mathematics and Computer Science, 1989.

25. T. Langholm, Partiality, Truth and Persistance, CSLI Lecture Notes No. 15, Stanford University, Stanford, 1988.

26. I.A. van Langevelde, A.W. Philipsen, J. Treur, Formal specification of compositional architectures, In: B. Neumann (ed.), Proc. 10th European Conference on Artificial Intelligence, ECAI'92, Wiley and Sons, 1992, pp. 272-276.

27. J.W. Lloyd, Foundations of logic programming, Springer Verlag, 1984.

28. P. Maes, D. Nardi (eds), Meta-level architectures and reflection, Elsevier Science Publishers, 1988.

29. Y.H. Tan and J. Treur, A bi-modular approach to nonmonotonic reasoning, In: De Glas, M., Gabbay, D. (eds.), Proc. World Congress on Fundamentals of Artificial Intelligence, WOCFAI'91, 1991, pp. 461-476.

30. Y.H. Tan and J. Treur, Constructive default logic and the control of defeasible reasoning, In: B. Neumann (ed.), Proc. 10th European Conference on Artificial Intelligence, ECAI'92, Wiley and Sons, 1992, pp. 299-303.

31. Y.H. Tan and J. Treur, Constructive default logic in a meta-level architecture, in: A Yonezawa, B.C. Smith (eds.), Proc. International Workshop on new Models in Software Architecture (IMSA) 1992, Reflection and Meta-level Architectures, 1992, pp. 184-189.

32. J. Treur, Completeness and definability in diagnostic expert systems, Proc. European Conference on Artificial Intelligence, ECAI'88, München, 1988, pp. 619-624.

33. J. Treur, On the use of reflection principles in modelling complex reasoning, International Journal of Intelligent Systems 6 (1991), pp. 277-294.

34. J. Treur, Declarative functionality descriptions of interactive reasoning modules, In: H. Boley, M.M. Richter (eds.), Processing Declarative Knowledge, Proc. of the International Workshop PDK'91, Lecture Notes in Artificial Intelligence, vol. 567, Springer Verlag, 1991, pp. 221-236.

35. J. Treur, P. Veerkamp, Explicit representation of design process knowledge, in: J.S. Gero (ed.), Artificial Intelligence in Design '92, Proc. AID'92, Kluwer Academic Publishers, 1992, pp. 677-696.

36. J. Treur and Th. Wetter (eds.), Formal Specification of Complex Reasoning Systems, Ellis Horwood, 1993, pp 282.

37. R.W. Weyhrauch, Prolegomena to a theory of mechanized formal reasoning, Artificial Intelligence 13 (1980), pp. 133-170.