Completeness of an Action Logic Featuring a δ-Operator for Timed Transition Systems

FERNANDO NÁUFEL DO AMARAL and EDWARD HERMANN HAEUSLER, *Department of Informatics, PUC–RJ (Catholic University of Rio de Janeiro), Brazil. E-mail: {fnaufel, hermann}@inf.puc-rio.br*

Abstract

This paper defines an action logic featuring an operator that denotes necessary conditions and postconditions of actions in a timed computational transition system. Three different semantics of the operator are discussed, and weak completeness is proved for one of them. It is also briefly shown how the action logic can be combined with a temporal logic to derive temporal properties of actions from the logical description of a real-time computational system.

Keywords: Modal Logic in Computing; Action Logic; Timed Transition Systems; Metric Temporal Logic

1 Introduction

In [10], Segerberg introduced the notion of a "bringing-it-about" δ operator, which, when applied to a given proposition q, denotes the set of actions that bring about the truth of q. In this paper, we discuss the use of a similar operator in a logical framework having computational transition systems as its semantics.

Segerberg's δ operator has led to an extension of dynamic logic containing elements from logics of action, where δq denotes actions that lead to states where q holds. In [3, 6], the operator was modified in two different ways: the first extension was introduced to reason about preconditions as well as postconditions, with the dyadic form $p\delta q$. The second extension, concerned with real-time applications, added a minimal and a maximal delay for the occurrence of each action: here, the action term $p_l\delta^u q$ denotes the actions that achieve q in less than u units of time, provided they were enabled in states satisfying p for at least l units of time.

In [3, 6], this last extension was dubbed RETOOL (Real-Time Object-Oriented Logic, in reference to the ultimate goal of achieving a formalization of real-time object-oriented software). The exact denotation of an action term $p\delta q$, however, has not been agreed upon. Two different semantics have been proposed for RETOOL, and this paper presents yet a third one:

- [3] defined $p\delta q$ to mean all actions that achieve q starting from states where p holds, with the further requirement that these actions be enabled in every state where p holds; we call this the *Enabling Condition Semantics* (ECS) of the δ operator.
- In [6], $p\delta q$ was defined as denoting all actions a such that, if a starts from a state

where p holds, then it achieves q; we call this the *Material Implication Semantics* (MIS) of the δ operator.

- In this paper, we present a definition of $p\delta q$ where this term means the set of actions achieving q from states where p holds; this is the *Necessary Condition Semantics* (NCS) of the δ operator.

The most promising semantic approaches regarding RETOOL as a logical language for describing computational transition systems seem to be the ECS and the NCS. In this paper we briefly discuss the reasons why the MIS may not be so suitable for modeling computational transition systems. We define a version of RETOOL corresponding to the NCS approach with a sound and (weakly) complete proof theory. This version of RETOOL may be useful as a logical language in a categorial framework for describing concurrent (possibly open) systems.[1] We conclude by briefly discussing the use of RETOOL combined with a temporal logic (particularly MTL [4]) for validating logic-based system descriptions.

2 Defining RETOOL

2.1 The Language

The primitive syntactic entities of RETOOL are *attribute symbols* (which, in this propositional version of the logic, are simple propositional letters), and *action symbols*. We denote the set of attribute symbols as A, and the set of action symbols as Γ.
The logic presupposes an infinite totally ordered set (TIME, $\leq$), with minimum 0. A constant ∞ is available, such that $\infty \notin$ TIME and $t \leq \infty, \forall t \in$ TIME. Also available are countably many constants, one for each member of TIME. An adequate theory of (TIME $\cup\{\infty\}$, $\leq$) is assumed to be contained in the logic; however, to make the presentation clearer, the axioms of this theory are not made explicit here. The other syntactic categories are:

- State Propositions (SP): $p ::= a \mid \neg p \mid p \rightarrow p'$, where $a \in A$;
- Action terms (AT): $t ::= g \mid p_l\delta^u q$, where $g \in \Gamma$, $p, q \in SP$, $l \in$ TIME, $u \in$ TIME $\cup\{\infty\}$, and $l \leq u$;
- Formulae: $\phi ::= a \mid t_1 \supset t_2 \mid \neg\phi \mid \phi \rightarrow \phi' \mid [t]\phi \mid [\,]p$, where $a \in A$, $t, t_1, t_2 \in AT$, and $p \in SP$.

There are also two unary function symbols, l and u, which can be applied to action terms to yield their time bounds (i.e., elements of TIME $\cup\{\infty\}$ – see below). The time bounds $l(g)$ and $u(g)$ of a primitive action symbol g are of an extra-logical nature; the time bounds $l(p_x\delta^y q)$ and $u(p_x\delta^y q)$ of an action term built with the δ operator are, respectively, the constants x and y, and can be considered abbreviations thereof.

2.2 Semantics

The semantics of RETOOL is defined over structures that are based on the notion of *timed transition systems* [8]: given a set A of attribute symbols and a set Γ of action symbols, a *timed frame* $\mathcal{F}$ for A and Γ is a sextuple $(W, \rightarrow, l, u, I, w_0)$, where

[1] We do not discuss this framework here. The reader is referred to [2, 5].

- W is a set of states;
- For each $g \in \Gamma$, $\xrightarrow{g} \subseteq W \times W$ is the transition relation for action g;
- l maps each $g \in \Gamma$ to an element $l(g) \in$ TIME;
- u maps each $g \in \Gamma$ to an element $u(g) \in$ TIME $\cup\{\infty\}$ such that $u(g) \geq l(g)$;
- $I : A \rightarrow 2^W$ is an interpretation of the attributes, where each $a \in A$ is assigned the set of worlds where a is true;
- w_0 is the initial state.

Every action $g \in \Gamma$ has a lower bound $l(g)$ and an upper bound $u(g)$. Intuitively, the lower bound defines the minimum delay that must be observed for the transition to take place (provided all the necessary conditions for the occurrence of such a transition are satisfied). The upper bound defines the maximum delay during which the transition must occur (again, provided all necessary conditions are satisfied). Formally, lower and upper bounds are defined through the use of the notion of *computation*:
A *timed state sequence* [8] for a timed frame is a pair $\rho =< \sigma, \mathrm{T} >$, where σ is an infinite sequence of states ($\sigma_i \in W$) and T is an infinite sequence of corresponding times ($\mathrm{T}_i \in$ TIME), satisfying:

- monotonicity: for all $i \geq 0$, either $\mathrm{T}_{i+1} = \mathrm{T}_i$, or ($\mathrm{T}_{i+1} > \mathrm{T}_i$ and $\sigma_{i+1} = \sigma_i$).
- progress: for every $t \in$ TIME, there is $i \geq 0$ such that $\mathrm{T}_i \geq t$.

A *computation* [8] over a timed frame is a timed state sequence $< \sigma, \mathrm{T} >$ such that

- σ is a computation of the underlying transition system, i.e., for every $i \geq 0$, there is a transition $\sigma(i)$ such that $\sigma_i \xrightarrow{\sigma(i)} \sigma_{i+1}$;
- (lower bound): for every $i \geq 0$ in the domain of σ, there is a $j \leq i$ such that $\mathrm{T}_i - \mathrm{T}_j > l(\sigma(i))$ and $\sigma(i)$ is enabled in every state σ_k for $j \leq k \leq i$.
- (upper bound): for every $g \in \Gamma$ and $i \geq 0$, there is $j \geq i$ with $\mathrm{T}_j - \mathrm{T}_i \leq u(g)$ such that either g is not enabled at σ_j or $g = \sigma(j)$.

The denotation of a state proposition p in a timed frame $\mathcal{F}$ is the set of states defined as follows:

- $[\![a]\!] = I(a)$;
- $[\![\neg p]\!]^{\mathcal{F}} = W \setminus [\![p]\!]^{\mathcal{F}}$;
- $[\![p \rightarrow p']\!]^{\mathcal{F}} = (W \setminus [\![p]\!]^{\mathcal{F}}) \cup [\![p']\!]^{\mathcal{F}}$.

The denotation of an action term t in a timed frame $\mathcal{F}$ is the set of transitions defined as follows (where $en(g)$ is the set of states where g is enabled):

- $[\![g]\!]^{\mathcal{F}} = \{(w, w') \mid w \xrightarrow{g} w'\}$;
- $[\![p_l \delta^u q]\!]^{\mathcal{F}} = \{(w, w') \mid \exists g \in \Gamma \ [(w \xrightarrow{g} w') \ \wedge \ en(g) \subseteq [\![p]\!]^{\mathcal{F}} \ \wedge$ $\forall v, v' ((v \xrightarrow{g} v') \Rightarrow (w' \in [\![q]\!]^{\mathcal{F}})) \ \wedge \ (l \leq l(g) \leq u(g) \leq u)]\}$

Finally, the satisfaction of a formula by a timed frame $\mathcal{F}$ at a state w is defined by:

- $\mathcal{F}, w \models p$ iff $w \in [\![p]\!]^{\mathcal{F}}$;
- $\mathcal{F}, w \models (t_1 \supset t_2)$ iff $[\![t_1]\!]^{\mathcal{F}} \subseteq [\![t_2]\!]^{\mathcal{F}}$;

- $\mathcal{F}, w \models \neg\phi$ iff not $\mathcal{F}, w \models \phi$;
- $\mathcal{F}, w \models \phi \rightarrow \phi'$ iff $\mathcal{F}, w \models \phi$ implies $\mathcal{F}, w \models \phi'$;
- $\mathcal{F}, w \models [t]\phi$ iff $\mathcal{F}, w' \models \phi$ for every w' such that $(w, w') \in [\![t]\!]^{\mathcal{F}}$;
- $\mathcal{F}, w \models [\,]p$ iff $\mathcal{F}, w_0 \models p$.

2.3 Axiomatization

The axiom schemes and rules of inference below comprise an adequate axiomatization of RETOOL. In what follows, Λ represents a set of RETOOL formulae, the derivability relation $\vdash$ is defined in the usual manner, and $enabled(t)$ is an abbreviation for the formula $\neg[t]\bot$ (see comments in the next section). For lack of space, we do not prove the soundness of this axiomatization, but it should be easy to see that the axioms and rules below are indeed sound for the Necessary Condition Semantics of RETOOL.

2.4 Comments

- "$\sqsupset$" is the *subsumption* operator. To say that action term t_1 subsumes action term t_2 is to say that every action denoted by t_1 is also denoted by t_2 (but not necessarily the other way around). Subsumption can be seen as a refinement on actions: the actions denoted by t_1 refine those denoted by t_2.
- We define, for an action term t, the set $en(t) = \{w \in W \mid \exists w'$ such that $(w, w') \in [\![t]\!]\}$. This is the set of all states where at least one of the actions in the denotation of t is enabled. Note that saying that a world w is a member of $en(t)$ is equivalent to saying that the formula $\neg[t]\bot$ is true in w. In fact, we define the abbreviation $enabled(t)$ to mean $\neg[t]\bot$.

The present work departs from the semantics of the δ operator given in [6]. There, $[\![p_l\delta^u q]\!]$ is defined as the set of all transitions (w, w') such that $w \in [\![p]\!]$ implies $w' \in [\![q]\!]$. This semantics mirrors the meaning of the Hoare triples $\{p\}g\{q\}$ (see [9]), namely that all terminating runs of program g starting in states satisfying p will end in states satisfying q (and there is no guarantee about runs that start in states not satisfying p). According to this view, p could be considered a precondition of program g, with q a relative postcondition.

Now, the semantics presented here requires that for (w, w') to be denoted by $p_l\delta^u q$, it must be the case that $w \xrightarrow{g} w'$ for some g such that $en(g) \subseteq [\![p]\!]$. The requirement that p must be true in all worlds where g is enabled forces us to see p as a necessary condition of the program represented by g. It is simply impossible to have runs of g starting in states not satisfying p.

This "necessary-condition" semantics (NCS) of the δ operator seems to us more appropriate to reason about timed transition systems as abstract models of complex real-time reactive systems. The "functionality" of an action, in the sense captured by Hoare triples, can still be expressed using action modalities. I.e., the Hoare triple $\{p\}g\{q\}$ is closely related to the RETOOL formula $p \rightarrow [g]q$.

(PC)	All axioms and rules of propositional calculus
(K)	$[t](\phi \rightarrow \psi) \rightarrow ([t]\phi \rightarrow [t]\psi)$ $[\,](p \rightarrow q) \rightarrow ([\,]p \rightarrow [\,]q)$
(I)	$[\,]p \rightarrow [t][\,]p$ $[t][\,]p \rightarrow (enabled(t) \rightarrow [\,]p)$ $[\,]\neg p \leftrightarrow \neg[\,]p$
(N)	$\dfrac{\Lambda \vdash \phi}{\Lambda \vdash [t]\phi} \quad \dfrac{\Lambda \vdash p}{\Lambda \vdash [\,]p}$
(δ)	$\dfrac{\Lambda \vdash enabled(t) \rightarrow p \quad \Lambda \vdash [t]q \quad \Lambda \vdash l \leq l(t) \leq u(t) \leq u}{\Lambda \vdash t \supset p_l\delta^u q}$
(S1)	$t \supset t$
(S2)	$(t_1 \supset t_2) \rightarrow ((t_2 \supset t_3) \rightarrow (t_1 \supset t_3))$
(S3)	$(t_1 \supset t_2) \rightarrow ([t_2]\phi \rightarrow [t_1]\phi)$
(NC)	$(t \supset p_l\delta^u q) \rightarrow (enabled(t) \rightarrow p)$
(Post)	$(t \supset p_l\delta^u q) \rightarrow ([t]q)$
(Bounds)	$(t \supset p_l\delta^u q) \rightarrow (enabled(t) \rightarrow l \leq l(t) \leq u(t) \leq u)$
(Global-$\supset$)	$(t_1 \supset t_2) \leftrightarrow [t](t_1 \supset t_2)$
(δ-Bounds)	$l(p_l\delta^u q) = l \quad u(p_l\delta^u q) = u$

Fig. 1. An Axiomatization of RETOOL

3 Weak Completeness

In order to show that the axiomatization is complete, we build a canonic model, whose worlds are maximally consistent sets of RETOOL formulae (see [7] for definitions and properties of maximally consistent sets), and whose transitions are determined from the contents of the worlds in the fashion described below. Since every consistent formula is a member of some maximally consistent set, and since satisfaction of a formula ϕ in a world w of the canonic model is equivalent to ϕ being a member of w (proven in the Coincidence Lemma), this ensures that *every* consistent formula will be satisfied in the canonic model.

3.1 Construction of the Canonic Model

The set A^c of attribute symbols of the canonic model is arbitrary: $A^c = \{a_1, a_2, ...\}$. For the construction of the set Γ^c of action symbols, we introduce the following definitions:

DEFINITION 3.1
For each $p \in SP$,[2] $\tilde{p} = \{r \mid \vdash p \leftrightarrow r\}$

DEFINITION 3.2
$\Gamma^c = \{\text{“}\tilde{p}_l \delta^u \tilde{q}\text{”} \mid p, q \in SP \setminus \tilde{\bot}, l \in \text{TIME}, u \in \text{TIME} \cup \{\infty\}\}$

We define here an action symbol for each quadruple (p, q, l, u) composed of satisfiable state propositions p and q (up to tautological equivalence) and time bounds l and u. In order to ensure that these action symbols have the desired meaning, we define the following set Σ of RETOOL formulae:

DEFINITION 3.3
$\Sigma = \{(\text{“}\tilde{p}_l \delta^u \tilde{q}\text{”} \supset p_l \delta^u q) \wedge (p_l \delta^u q \supset \text{“}\tilde{p}_l \delta^u \tilde{q}\text{”}) \mid p, q \in SP \setminus \tilde{\bot}, \quad l \in \text{TIME}, \quad u \in \text{TIME} \cup \{\infty\}\}$

Note that in every model of Σ, each action symbol “$\tilde{p}_l \delta^u \tilde{q}$” will function as a “witness” to action term $p_l \delta^u q$, the denotation of the symbol corresponding exactly to the denotation of the term.
More extra-logical information is necessary to build the canonic model: we must specify the initial state through a set Θ of formulae. If we want state propositions $p_1, p_2, ...$ to be true at the initial state w_0, we define $\Theta = \{[\,]p_1, [\,]p_2, ...\}$.[3] Obviously, the set $\{p_1, p_2, ...\}$ must be consistent.
We also introduce some special notation to refer to “demodalized” formulae, where w is a set of formulae and t is any action term:

DEFINITION 3.4
$w \setminus [t] = \{\phi \mid [t]\phi \in w\}$

DEFINITION 3.5 (The canonic model)
Given the sets A^c, Σ, Θ and Γ^c, we define the canonic model as $M^c = (W^c, \{\xrightarrow{g} \mid g \in \Gamma^c\}, l^c, u^c, I^c, w_0^c)$, where

W^c	$= \{\Phi \mid \Phi$ is a maximally consistent set containing $\Sigma \cup \Theta\}$.
$\xrightarrow{\text{“}\tilde{p}_l \delta^u \tilde{q}\text{”}}$	$= \{(w, w') \in W^c \times W^c \mid w \setminus [\text{“}\tilde{p}_l \delta^u \tilde{q}\text{”}] \subseteq w'\}$
l^c	is such that $l^c(\text{“}\tilde{p}_l \delta^u \tilde{q}\text{”}) = l$ for every “$\tilde{p}_l \delta^u \tilde{q}$” $\in \Gamma^c$.
u^c	is such that $u^c(\text{“}\tilde{p}_l \delta^u \tilde{q}\text{”}) = u$ for every “$\tilde{p}_l \delta^u \tilde{q}$” $\in \Gamma^c$.
I^c	is such that $I^c(p) = \{w \in W^c \mid p \in w\}$ for every $p \in SP$.
w_0^c	is some world in W^c satisfying $\Theta \setminus [\,]$.

Central to the completeness proof is the Coincidence Lemma:

$$\forall w \in W^c,\ \forall \phi: \quad \phi \in w \quad \Leftrightarrow \quad M^c, w \models \phi$$

[2] We refer to the set of state propositions as SP, and to the set of action terms as AT.
[3] Actually, this leads to a *class* of canonic models, one model for each choice of Θ.

This is proved by induction over the formation of ϕ. We sketch the proof of only one case here, using the following lemmas:[4]

LEMMA 3.6
For all $w \in W^c$, for all $p, q \in SP \setminus \tilde{\bot}$, for all $l \in$ TIME , $u \in$ TIME $\cup \{\infty\}$, for all $t \in AT$, and for all formulae ϕ:

$$\begin{array}{lcl} w \vdash t \supset p_l\delta^u q & \Leftrightarrow & w \vdash t \supset \text{“}\tilde{p}_l\delta^u\tilde{q}\text{”} \\ w \vdash p_l\delta^u q \supset t & \Leftrightarrow & w \vdash \text{“}\tilde{p}_l\delta^u\tilde{q}\text{”} \supset t \\ w \vdash [p_l\delta^u q]\phi & \Leftrightarrow & w \vdash [\text{“}\tilde{p}_l\delta^u\tilde{q}\text{”}]\phi \end{array}$$

LEMMA 3.7
For all $w \in W^c$, for all $p, q \in SP \setminus \tilde{\bot}$, for all $l \in$ TIME , $u \in$ TIME $\cup \{\infty\}$:

$$w \in en(p_l\delta^u q) \quad \Leftrightarrow \quad w \vdash enabled(p_l\delta^u q)$$

By lemma 3.6, the equivalence also holds for the formula $enabled(\text{“}\tilde{p}_l\delta^u\tilde{q}\text{”})$.

LEMMA 3.8
In the canonic model, $[\![\text{“}\tilde{p}_l\delta^u\tilde{q}\text{”}]\!] = [\![p_l\delta^u q]\!]$.

The subsumption case of the Coincidence Lemma: $\phi = t_1 \supset t_2$

By lemmas 3.6 and 3.8, we may restrict ourselves to the case where $t_1 = p_l\delta^u q$ and $t_2 = r_m\delta^n s$:

($\Rightarrow$) Suppose $p_l\delta^u q \supset r_m\delta^n s \in w$. Let $(w, w') \in [\![p_l\delta^u q]\!]$. Then

$$\begin{array}{lcl} p_l\delta^u q \supset r_m\delta^n s \in w & \Rightarrow & \text{(by axiom (S3))} \\ \forall\phi \quad [r_m\delta^n s]\phi \to [p_l\delta^u q]\phi \quad \in w & \Rightarrow & \text{(as } w \text{ is maximally consistent)} \\ \forall\phi \quad [r_m\delta^n s]\phi \in w \quad \Rightarrow \quad [p_l\delta^u q]\phi \in w & \Rightarrow & \text{(as } w \setminus [p_l\delta^u q] \subseteq w'\text{)} \\ w \setminus [r_m\delta^n s] \subseteq w' & \Rightarrow & \text{(by the definition of } M^c\text{)} \\ (w, w') \in [\![r_m\delta^n s]\!] & & \end{array}$$

($\Leftarrow$) Suppose $w \models p_l\delta^u q \supset r_m\delta^n s$. Then, for every $w \in W^c$,

$$\begin{array}{lcl} [\![p_l\delta^u q]\!] \subseteq [\![r_m\delta^n s]\!] & \Rightarrow & \\ w \models enabled(p_l\delta^u q) \quad \Rightarrow \quad w \models r & \Leftrightarrow & \text{(by lemma 3.7 and the definition of } I^c\text{)} \\ w \vdash enabled(p_l\delta^u q) \quad \Rightarrow \quad w \vdash r & \Leftrightarrow & \text{(as } w \text{ is maximally consistent)} \\ w \vdash enabled(p_l\delta^u q) \to r & \Leftrightarrow & \text{(as } w \text{ is arbitrary containing } \Sigma \cup \Theta\text{)} \\ \Sigma \cup \Theta \vdash enabled(p_l\delta^u q) \to r & & \end{array}$$

As for the postcondition, we have that, for all $w \in M^c$,

$$\begin{array}{lcl} [\![p_l\delta^u q]\!] \subseteq [\![r_m\delta^n s]\!] & \Rightarrow & \\ w \models [p_l\delta^u q]s & \Rightarrow & \text{(by lemma 3.8)} \\ w \vdash [p_l\delta^u q]s & \Leftrightarrow & \text{(as } w \text{ is arbitrary containing } \Sigma \cup \Theta\text{)} \\ \Sigma \cup \Theta \vdash [p_l\delta^u q]s & & \end{array}$$

As for the time bounds, we have that $m \leq l \leq u \leq n$. There are derived rules that state that necessary conditions, postconditions and time bounds can be weakened. Therefore, $\Sigma \cup \Theta \vdash p_l\delta^u q \supset r_m\delta^n s$, and we conclude that, as any $w \in M^c$ contains $\Sigma \cup \Theta$, it is the case that $w \vdash p_l\delta^u q \supset r_m\delta^n s$.

[4] Detailed proofs of these and all other relevant lemmas can be found in [1].

4 Abstracting Temporal Properties of Systems

Since the semantics of the time-bounded δ operator is given in terms of computations, it is only natural that we investigate the combination of RETOOL with a linear-time temporal logic, whose models are also computations. In these concluding remarks, we briefly describe the nature of this combination, which has been studied for the MIS semantics of RETOOL in [6] and can be easily adapted to the NCS semantics presented in this paper.

The temporal logic chosen was MTL (see [4]), which features time-bounded temporal operators $\mathbf{X}_{Rc}$ (next state) and $\mathbf{U}_{Rc}$ (until), where $c \in \text{TIME} \cup \{\infty\}$ and R is a relation on TIME. We extend this language by adding RETOOL's action terms as primitive propositions and stipulate that an action term t is true at step i of a computation iff the denotation of t contains the action responsible for the transition taken at step i.

The following are examples (taken from [6] and modified to fit our NCS semantics) of proof rules relating RETOOL and MTL:

$$\frac{\text{enabled}(t) \rightarrow r \qquad t \supset p_0 \delta^\infty q}{t \rightarrow r \wedge \mathbf{X}_{=0} q}$$

This rule states that an action denoted by t (with p as a necessary condition and q as a postcondition) establishes q in the next state whenever this action is responsible for the transition to the next state. In compliance with the definition of computation, the transition does not take any time.

$$\frac{p \rightarrow \neg\text{enabled}(t) \qquad t \supset \top_x \delta^\infty \top}{p \rightarrow \mathbf{G}_{\leq x} \neg t}$$

This rule captures a safety property concerning the lower bound of an action: if the truth of p means that action t is disabled, and t has lower bound x, we conclude that, p holding in the current state, t will not be taken for at least x time units (the $\mathbf{G}_{\leq x}$ bounded temporal operator meaning "during the next x time units").

Additional rules and an example of a detailed system description can be found in [6] for the MIS semantics. They can be easily adapted to the NCS semantics we employ here.

5 Concluding Remarks

In this paper, we have shown how Segerberg's δ operator can be extended to define RETOOL, an action logic meant to reason about computational timed transition systems. Three alternative semantics have been considered for action terms built with this operator, and a weak completeness proof has been sketched for one of these semantics (the NCS).

It was also indicated how the combination of RETOOL with a temporal logic can be useful in deriving properties of actions from system descriptions.

Work in progress includes the study of the ECS as an alternative semantics, with soundness and completeness proofs, and the comparison of the ECS and NCS approaches in the specification of real-time computational systems.

Acknowlegdements

We would like to thank Tom Maibaum and José Fiadeiro for the many fruitful discussions about RETOOL.

References

[1] Amaral, F.N., "RETOOL: Uma Lógica de Ações para Sistemas de Transição Temporizados", M.Sc. Dissertation, Dept. of Informatics, PUC-RJ, Brazil, 2000.

[2] Bicarregui, J., Lano, K. and Maibaum, T., "Towards a Compositional Interpretation of Object Diagrams", in *Proc IFIP Working Conference on Algorithmic Languages and Calculi*, Chapman and Hall, 1997.

[3] Carvalho, S., Fiadeiro, J. e Haeusler, E.H., "A Formal Approach to Real-Time Object-Oriented Software", in *Proc. 22nd IFAC/IFIP Workshop on Real-Time Programming WRTP'97*, Elsevier 1997.

[4] Chang, E., *Compositional Verification of Reactive and Real-Time Systems*, PhD Thesis, Stanford University, 1995.

[5] Fiadeiro, J. and Maibaum, T., "Temporal Theories as Modularization Units for Concurrent Systems Specification", in *Formal Aspects of Computing* 4(3), 1992.

[6] Fiadeiro, J. and Haeusler, E.H., "Bringing It About On Time (Extended Abstract)", in *Proc. I IMLLAI, Fortaleza, CE, Brazil*, 1998.

[7] Goldblatt, R., *Logics of Time and Computation*, CSLI Lecture Notes 7, CSLI, 1992.

[8] Henzinger, T., Manna, Z., e Pnuelli, A., "Timed Transition Systems", in *Real Time: Theory in Practice* (J.W. de Bakker, C. Huizing, W.P. de Roever e G. Hozenberg (eds.)), LNCS 600, Springer-Verlag, 1992.

[9] Hoare, C.A.R., "An Axiomatic Basis for Computer Programming", in *Comm. ACM* 12, 1967.

[10] Segerberg, K., "Bringing It About", in *Journal of Philosophical Logic* 18, 1989.