In: Proc. of the workshop on Design and Analysis of Real-Time Systems (DARTS'95), Université libre de Bruxelles, Nov. 95

An Extended LOTOS for the design of Real-Time Systems

Luc Léonard and Guy Leduc
Research Assistant and Research Associate of the
National Fund for Scientific Research (Belgium)
Université de Liège, Institut d'Electricité Montefiore, B 28, B-4000 Liège 1, Belgium
Tel: + 32 4 3662697 Fax: + 32 4 3662989 E-mail: leonard@montefiore.ulg.ac.be

1. Introduction

We give in the following a brief presentation of ET-LOTOS [LéL 95a, LéL 95b]. ET-LOTOS extends with quantative time the formal description technique LOTOS [ISO 8807]. Other proposals for a "time extended" LOTOS exist. Let us mention [QMF 94] and [BLT 94]. ET-LOTOS serves as basis for the time extension part of E-LOTOS, the new standard for LOTOS currently developed within ISO (ISO/IEC JTC1/SC21).

We assume in the sequel that the reader has a basic knowledge of the syntax and the semantics of LOTOS.

2. Formal semantics and properties of ET-LOTOS

2.1. Datatypes and time domain

In ET-LOTOS, like in LOTOS, datatypes are described in the Abstract Datatype language ACT ONE, that has an initial semantics.

The time domain, denoted D, is defined as the set of values of a given data sort `time` ($D = Q(\mathtt{time})$). Its definition is left free to the will of the specifier provided that the following elements be defined.

- A total order relation represented by ">".
- An element $0 \in D$ such that: $\forall\ r \in D\colon r \neq 0 \Rightarrow r > 0$
- An element $\infty \in D$ such that: $\forall\ r \in D\colon r \neq \infty \Rightarrow \infty > r$
- A commutative and associative operation "$+ : D, D \rightarrow D$" such that:

  $\forall\ r, r1 \in D\colon r > r1 \Leftrightarrow \exists\ r' > 0 \bullet (r' + r1) = r$

  $\forall\ r, r1 \in D\colon r > 0 \text{ and } r1 \neq \infty \Rightarrow r + r1 > r1$

  $\forall\ r \in D\colon r + 0 = r$

  $\forall\ r \in D\colon r + \infty = \infty$

The relations "≤", and "-" can be derived easily as follows :

$\forall\ r, r1 \in D \bullet r \leq r1 \Leftrightarrow (r < r1 \vee r1 = r))$

$\forall\ r, r1, r2 \in D \bullet r1 \leq r \Rightarrow (r - r1 = r2 \Leftrightarrow r1 + r2 = r)$

$\forall\ r, r1 \in D \bullet r \leq r1 \Rightarrow r - r1 = 0$

In particular, the time domain can be dense as well as discrete, but to be able to give the operational semantics of ET-LOTOS in terms of Labelled Transition Systems (LTS), it must be countable, such as the rational numbers.

2.2 Notations

The following notations hold for the remainder of the paper. G denotes the countable set of common observable gates. $L = G \cup \{\delta\}$ denotes the alphabet of observable gates where δ is the special action denoting successful termination ($\delta \notin G$). δ does not appear explicitly in the syntax of LOTOS. S denotes the set of sorts, V denotes the set of ground terms in the initial algebra associated with the ACT ONE specification: $V = \biguplus_{s \in S} Q(s)$. $CL = L \times V^*$ denotes the set of observable actions. $A = CL \cup \{i\}$ denotes the alphabet of actions, where the symbol `i` is reserved for the unobservable internal action ($i \notin L$). `g` (resp. `a`) denotes an element of G (resp. A): $g \in G$, $a \in A$. $g v_1 \ldots v_n$ and $\delta v_1 \ldots v_n$ denote elements of CL, with the v_i's $\in V$. Capital Greek letters such as Γ will be used to denote subsets of G. D denotes the countable time domain which is the alphabet of time actions. $D_{0\infty} = D - \{0,\infty\}$.

2.3 Syntax of the behaviour part of ET-LOTOS

The collection of ET-LOTOS behaviour expressions is defined by the following BNF expressions. In these expressions, $\tilde{X}$ represents a vector of process names, `SP` is a selection predicate, the e_i's represent a term[1] `tx`, the o_i's represent either `?x:s` (with `x` a variable of sort `s`) or `!tx` (with `tx` a ground term), the x_i's (resp. tx_i's) are variables (resp. ground terms) of sorts s_i's, $d \in D$ and in `@t`, `t` is a variable of sort `time`. The new features are printed in italics:

$$P ::= Q \ \texttt{where} \ \tilde{X} := \tilde{Q}$$ [2]

$$\begin{array}{ll} Q ::= & \texttt{stop} \mid \texttt{exit}(e_1,\ldots e_n)\mathit{\{d\}} \mid g o_1 \ldots o_n \mathit{@t}[SP];Q \mid i\mathit{@t\{d\}};Q \mid \mathit{\Delta^d}\, Q \mid Q[\,]Q \mid Q|[\Gamma]|Q \mid \\ & \texttt{hide}\ \Gamma\ \texttt{in}\ Q \mid Q\texttt{>>accept}\ x_1{:}s_1,\ldots x_n{:}s_n\ \texttt{in}\ Q \mid Q[>Q \mid X \mid [SP] \texttt{->} Q \mid \\ & \texttt{let}\ x_1{=}tx_1,\ \ldots\ x_n{=}tx_n\ \texttt{in}\ Q \mid \texttt{choice}\ x_1{:}s_1,\ \ldots\ x_n{:}s_n\ [\,]\ Q \mid \mathit{inf}\ |||\ P \end{array}$$

Remark: in $g o_1 \ldots o_n @t[SP];Q$ we let both `@t` and `[SP]` be optional, and use the convention that, if omitted, `[SP] = [true]`. In `i@t{d};Q`, both `@t` and `{d}` are optional. If omitted, $d = 0$. Similarly `{d}` is optional in `exit{d}`, and `exit` means implicitly `exit{∞}`.

The binding powers of the operators are like in LOTOS. For the new operators, Δ^d has the same power as action-prefix and `inf |||` the same as `choice` $x_1{:}s_1,\ldots x_n{:}s_n$ `[]`.

An additional shorthand notation: We define the notation $g o_1 \ldots o_n\{d\};Q$, for $g o_1 \ldots o_n @t[t \le d];Q$, provided that `t` be fresh in `Q`. Under the same restriction, we also introduce the notation

[1] This term can be:‘`any s`’ (with $s \in S$)
[2] For convenience, we suppose, without lack of generality, that there is a single where-clause that gathers all the process declarations of the specification.

$g o_1 \ldots o_n \{d_1, d_2\}; P$ to mean $g@t[d_1 \leq t \leq d_2]; P$. The meaning of these rewritings will become clear in the next section.

2.4 Semantics of ET-LOTOS

The operational semantics of ET-LOTOS, presented in the following, is of the so-called "time/actions" type. This means that the occurrence of actions and the passing of time are considered as separate concerns, each one being described by a dedicated set of rules.

2.4.1 Notations

P, P', Q, Q' denote ET-LOTOS behaviour expressions.

$P \xrightarrow{a} P'$, with $a \in A$, means that process P may engage in action a and, after doing so, behave like process P'. $P \xrightarrow{g}$ means $\exists\, P', a \bullet P \xrightarrow{a} P' \wedge \text{name}(a) = g$. $P \not\xrightarrow{g}$ means $\neg\, (P \xrightarrow{g})$ i.e. P cannot perform an action on gate g. $P \xrightarrow{d} P'$, with $d \in D_{0\infty}$, means that process P may idle (i.e. not execute any action in A) during a period of d units of time and, after doing so, behave like process P'. $P \not\xrightarrow{d}$, with $d \in D_{0\infty}$, means that $\nexists\, P' \bullet P \xrightarrow{d} P'$, i.e. P cannot idle during a period of d units of time. In these expressions, it is required that P and P' be closed, i.e. they do not contain free variables.

2.4.2 Inference rules

In the following inference rules, $d \in D_{0\infty}$, $d1 \in D$, $d' \in D_\infty$, $g \in G$ and $a \in A$.

We introduce a process, denoted `block`, which has no axiom and no inference rules. This process cannot perform any action and blocks the progression of time.

Inaction

(S) $\quad \text{stop} \xrightarrow{d} \text{stop}$

Remark that `stop` cannot perform any action but can idle.

Exit

(Ex1) $\quad \text{exit}(e_1, \ldots e_n)\{d1\} \xrightarrow{\delta v_1 \ldots v_n} \text{stop}$

where $v_i = [t_i]$ if $e_i = t_i$ (a ground term)

$v_i \in Q(s_i) = \{[t] \mid t \text{ is a ground term of sort } s_i\}$ if $e_i = \text{any } s_i$

(Ex2) $\quad \text{exit}(e_1, \ldots e_n)\{d1+d\} \xrightarrow{d} \text{exit}(e_1, \ldots e_n)\{d1\}$

(Ex3) $\quad \text{exit}(e_1, \ldots e_n)\{d1\} \xrightarrow{d} \text{stop} \qquad (d > d1)$

The `{d1}` attribute is called the life reducer. Its role is to restrict the time period during which the process can terminate successfully: `exit{d1}` can only perform δ during the next `d1` time units. If `exit{d1}` has not performed δ yet after `d1` time units, it is too late and the process turns into `stop` (rule Ex3).

Observable action-prefix

(AP1) $\texttt{g}o_1 \ldots o_n\texttt{@t[SP];P} \xrightarrow{\texttt{g}v_1\ldots v_n} [v_1/o_1,\ldots v_m/o_m,\ \texttt{0/t]P}$

if $\vdash [v_1/o_1,\ldots v_m/o_m,\ \texttt{0/t]SP}$

$v_i = \texttt{[w]}$ if $o_i = \texttt{!w}$

$v_i \in \texttt{Q(s)} = \{\texttt{[w]} \mid \texttt{w}$ is a ground term of sort $\texttt{s}\}$ if $o_i = \texttt{?x:s}$

and where $v_i/o_i = v_i/\texttt{x}$ if $o_i = \texttt{?x:s}$

v_i/o_i is void if $o_i = \texttt{!w}$

(AP2) $\texttt{g}o_1 \ldots o_n\texttt{@t[SP];P} \xrightarrow{d} \texttt{g}o_1 \ldots o_n\texttt{@t[[t+d/t]SP];[t+d/t]P}$

In `@t`, `t` is a variable of sort `time`. This variable is used to measure the delay actions were being offered on `g` when one occurred. When an action occurs (rule AP1), `t` is instantiated. Instantiating `t` by `0` is logical: $\texttt{g}o_1 \ldots o_n\texttt{@t[SP];P}$ describes a process at a given instant and the counting of `t` starts at that instant. So, `t` is still at `0` if the process immediately does an action on gate `g`. The way the value of `t` is kept up to date if $\texttt{g}o_1 \ldots o_n\texttt{@t[SP];P}$ idles is defined by AP2.

The `t` variable can appear in the selection predicate `SP`, if there is one. The conditions joined with AP1 express that the only possible instantiations for the attributes of `g` are the ones that make `SP` true at that instant.

Internal action-prefix

(I1) $\texttt{i@t\{d1\};P} \xrightarrow{i} \texttt{[0/t]P}$ (I2) $\texttt{i@t\{d1+d\};P} \xrightarrow{d} \texttt{i@t\{d1\};[t+d/t]P}$

There is no rule like Ex3 for the internal action-prefix. `i@t{d1};P` cannot idle more than `d1` time units. If it reaches this limit, time is blocked. The only solution left is to accomplish `i`. This means that, in Timed Extended LOTOS, the occurrence of `i` is compulsory. The semantics of `i@t{d1};P` is that `i` *shall* occur during the next `d1` time unitsr[3]. On the other hand, the semantics of `exit(d1)` is that δ *may* occur within the next `d1` time units.

Delay prefixing

(D1) $\dfrac{\texttt{P} \xrightarrow{a} \texttt{P}'}{\Delta^{0}\ \texttt{P} \xrightarrow{a} \texttt{P}'}$ (D2) $\Delta^{\texttt{d1+d}}\ \texttt{P} \xrightarrow{d} \Delta^{\texttt{d1}}\ \texttt{P}$

(D3) $\dfrac{\texttt{P} \xrightarrow{d} \texttt{P}'}{\Delta^{\texttt{d1}}\ \texttt{P} \xrightarrow{d+d1} \texttt{P}'}$

$\Delta^{\texttt{d}}\texttt{;P}$ expresses that `P` will be delayed by `d` time units.

Choice

(Ch1) $\dfrac{\texttt{P} \xrightarrow{a} \texttt{P}'}{\texttt{P[]Q} \xrightarrow{a} \texttt{P}'}$ (Ch1') $\dfrac{\texttt{Q} \xrightarrow{a} \texttt{Q}'}{\texttt{P[]Q} \xrightarrow{a} \texttt{Q}'}$ (Ch2) $\dfrac{\texttt{P} \xrightarrow{d} \texttt{P}',\ \texttt{Q} \xrightarrow{d} \texttt{Q}'}{\texttt{P[]Q} \xrightarrow{d} \texttt{P}'\texttt{[]Q}'}$

Remark rule Ch2: the passing of time does not resolve a choice. Rule Ch2 also states that both operands evolve in time at the same pace.

[3] Of course, in a choice context, the occurrence of i could be prevented by another offered action.

Generalized choice

The semantics of `choice` $x_1{:}s_1,\dots x_n{:}s_n[\,]P$ is defined via an auxiliary operator, denoted $\mathtt{Achoice(d)}\ x_1{:}s_1,\dots x_n{:}s_n[\,]P$, where $d \in D_\infty$. `Achoice` stands for `AgedChoice`. By definition, $\mathtt{choice}\ x_1{:}s_1,\dots x_n{:}s_n[\,]P = \mathtt{Achoice(0)}\ x_1{:}s_1,\dots x_n{:}s_n[\,]P$.

(GC1) $$\frac{[tx_1/x_1,\ \dots\ tx_n/x_n]P \xrightarrow{a} P'}{\mathtt{Achoice(0)}\ x_1{:}s_1,\ \dots\ x_n{:}s_n[\,]P \xrightarrow{a} P'}$$

(GC2) $$\frac{[tx_1/x_1,\ \dots\ tx_n/x_n]P \xrightarrow{d} P'',\ P'' \xrightarrow{a} P'}{\mathtt{Achoice(d)}\ x_1{:}s_1,\ \dots\ x_n{:}s_n[\,]P \xrightarrow{a} P'} \qquad \text{if } d > 0$$

where the tx_i are ground terms with $[tx_i] \in Q(s_i)$

(GC3) $$\frac{[tx_1/x_1,\dots tx_n/x_n]P \xrightarrow{d+d'} \quad \forall \langle tx_1,\dots tx_n\rangle \bullet [tx_i] \in Q(s_i),\ i = 1,\dots n}{\mathtt{Achoice(d')}\ x_1{:}s_1,\ \dots\ x_n{:}s_n[\,]P \xrightarrow{d} \mathtt{Achoice(d+d')}\ x_1{:}s_1,\dots x_n{:}s_n[\,]P}$$

Parallel composition

(PC1) $$\frac{P \xrightarrow{a} P'}{P|[\Gamma]|Q \xrightarrow{a} P'|[\Gamma]|Q} \qquad (\mathrm{name}(a) \notin \Gamma \cup \{\delta\})$$

(PC3) $$\frac{P \xrightarrow{d} P',\ Q \xrightarrow{d} Q'}{P|[\Gamma]|Q \xrightarrow{d} P'|[\Gamma]|Q'}$$

(PC1') $$\frac{Q \xrightarrow{a} Q'}{P|[\Gamma]|Q \xrightarrow{a} P|[\Gamma]|Q'} \qquad (\mathrm{name}(a) \notin \Gamma \cup \{\delta\})$$

(PC2) $$\frac{P \xrightarrow{a} P',\ Q \xrightarrow{a} Q'}{P|[\Gamma]|Q \xrightarrow{a} P'|[\Gamma]|Q'} \qquad (\mathrm{name}(a) \in \Gamma \cup \{\delta\})$$

Infinite parallel composition

(IP1) $$\frac{P \xrightarrow{a} P'}{\mathtt{inf}|||P \xrightarrow{a} P'\ |||\ (\mathtt{inf}|||P)}$$

(IP2) $$\frac{P \xrightarrow{d} P'}{\mathtt{inf}|||P \xrightarrow{d} \mathtt{inf}|||P'}$$

`inf|||P` corresponds to an infinity of occurrences of `P` evolving in parallel. In ET-LOTOS, such a behaviour cannot be described by a recursive process like `Ps:= P|||Ps`, because unguarded recursions block time (see [LéL 95b]).

Hide

(H1) $$\frac{P \xrightarrow{a} P'}{\mathtt{hide}\ \Gamma\ \mathtt{in}\ P \xrightarrow{a} \mathtt{hide}\ \Gamma\ \mathtt{in}\ P'} \qquad (a \notin \Gamma)$$

(H2) $$\frac{P \xrightarrow{a} P'}{\mathtt{hide}\ \Gamma\ \mathtt{in}\ P \xrightarrow{i} \mathtt{hide}\ \Gamma\ \mathtt{in}\ P'} \qquad (a \in \Gamma)$$

(H3) $$\frac{P \xrightarrow{d} P',\ \forall g \in \Gamma \bullet (P \overset{g}{\not\to} \wedge \forall P''\ \forall d'<d \bullet (P \xrightarrow{d'} P'' \Rightarrow P'' \overset{g}{\not\to}))}{\mathtt{hide}\ \Gamma\ \mathtt{in}\ P \xrightarrow{d} \mathtt{hide}\ \Gamma\ \mathtt{in}\ P'}$$

Rule (H3) expresses the *maximal progress* principle adopted for ET-LOTOS. This principle states that the hidden events must occur as soon as possible. So, the process can only idle if no hidden action is possible.

Enabling

(En1) $$\frac{P \xrightarrow{a} P'}{P \gg \mathtt{accept}\ x_1{:}s_1,\dots x_n{:}s_n\ \mathtt{in}\ Q \xrightarrow{a} P' \gg \mathtt{accept}\ x_1{:}s_1,\dots x_n{:}s_n\ \mathtt{in}\ Q} \qquad (\mathrm{name}\{a\} \neq \delta)$$

(En2) $$\frac{\mathtt{P} \xrightarrow{\delta v_1 \ldots v_n} \mathtt{P'}}{\mathtt{P>>accept}\ x_1{:}s_1,\ldots x_n{:}s_n\ \mathtt{in}\ \mathtt{Q} \xrightarrow{i} [v_1/x_1,\ldots v_n/x_n]\mathtt{Q}} \quad \forall j \le n \bullet v_j \in Q(s_j)$$

(En3) $$\frac{\mathtt{P} \xrightarrow{d} \mathtt{P'},\ \mathtt{P} \not\xrightarrow{\delta},\ \forall \mathtt{P''} \forall d'<d \bullet (\mathtt{P} \xrightarrow{d'} \mathtt{P''} \Rightarrow \mathtt{P''} \not\xrightarrow{\delta})}{\mathtt{P>>accept}\ x_1{:}s_1,\ \ldots\ x_n{:}s_n\ \mathtt{in}\ \mathtt{Q} \xrightarrow{d} \mathtt{P'>>accept}\ x_1{:}s_1,\ \ldots\ x_n{:}s_n\ \mathtt{in}\ \mathtt{Q}}$$

The occurrence of δ is hidden by the enabling operator. According to the maximal progress principle, it must occur as soon as possible.

Disabling

(Di1) $$\frac{\mathtt{P} \xrightarrow{a} \mathtt{P'}}{\mathtt{P[>Q} \xrightarrow{a} \mathtt{P'[>Q}} \quad (\mathrm{name}(a) \neq \delta)$$

(Di2) $$\frac{\mathtt{Q} \xrightarrow{a} \mathtt{Q'}}{\mathtt{P[>Q} \xrightarrow{a} \mathtt{Q'}}$$

(Di3) $$\frac{\mathtt{P} \xrightarrow{a} \mathtt{P'}}{\mathtt{P[>Q} \xrightarrow{a} \mathtt{P'}} \quad (\mathrm{name}(a) = \delta)$$

(Di4) $$\frac{\mathtt{P} \xrightarrow{d} \mathtt{P'},\ \mathtt{Q} \xrightarrow{d} \mathtt{Q'}}{\mathtt{P[>Q} \xrightarrow{d} \mathtt{P'[>Q'}}$$

Guard

(G1) $$\frac{\mathtt{P} \xrightarrow{a} \mathtt{P'}}{\mathtt{[SP]\text{->}P} \xrightarrow{a} \mathtt{P'}} \quad \text{if DS} \vdash \text{SP}$$

(G2) $$\frac{\mathtt{P} \xrightarrow{d} \mathtt{P'}}{\mathtt{[SP]\text{->}P} \xrightarrow{d} \mathtt{P'}} \quad \text{if DS} \vdash \text{SP}$$

(G3) $$\mathtt{[SP]\text{->}P} \xrightarrow{d} \mathtt{stop} \quad \text{if} \neg \text{DS} \vdash \text{SP}$$

Let

(L1) $$\frac{[tx_1/x_1,\ldots tx_n/x_n]\ \mathtt{P} \xrightarrow{a} \mathtt{P'}}{\mathtt{let}\ x_1{=}tx_1,\ldots x_n{=}tx_n\ \mathtt{in}\ \mathtt{P} \xrightarrow{a} \mathtt{P'}}$$

(L2) $$\frac{[tx_1/x_1,\ldots tx_n/x_n]\ \mathtt{P} \xrightarrow{d} \mathtt{P'}}{\mathtt{let}\ x_1{=}tx_1,\ldots x_n{=}tx_n\ \mathtt{in}\ \mathtt{P} \xrightarrow{d} \mathtt{P'}}$$

Process instantiation

(In1) $$\frac{[g_1/h_1,\ldots g_n/h_n]\ \mathtt{P} \xrightarrow{a} \mathtt{P'},\ \mathtt{Q}[h_1,\ldots h_n]{:=}\mathtt{P}}{\mathtt{Q}[g_1,\ldots g_n] \xrightarrow{a} \mathtt{P'}}$$

(In2) $$\frac{[g_1/h_1,\ldots g_n/h_n]\ \mathtt{P} \xrightarrow{d} \mathtt{P'},\ \mathtt{Q}[h_1,\ldots h_n]{:=}\mathtt{P}}{\mathtt{Q}[g_1,\ldots g_n] \xrightarrow{d} \mathtt{P'}}$$

Let us outline some interesting features of the semantic rules defined above:

- The LOTOS rules are kept unchanged.
- The alphabet $\mathtt{A}$ of actions is kept as is (e.g. no additional time stamps in action labels). It is just extended with time actions from a separate set $\mathtt{D}$.

2.5. Properties

ET-LOTOS exhibits many interesting properties (the proofs can be found in [LéL 95b]):

- The operational semantics of ET-LOTOS is consistent.
- Time transitions are deterministic: $\forall \mathtt{P} \bullet (\mathtt{P} \xrightarrow{d} \mathtt{P'} \wedge \mathtt{P} \xrightarrow{d} \mathtt{P''}) \Rightarrow \mathtt{P'} = \mathtt{P''}$.
- Time transitions are closed under the relation $\le$: $\mathtt{P} \xrightarrow{d} \Rightarrow \forall d' \in]0,d] \bullet \mathtt{P} \xrightarrow{d'}$.
  Furthermore, $\mathtt{P} \xrightarrow{d} \mathtt{P'} \Rightarrow \forall d' \in]0,d[\bullet \exists d'' \bullet \mathtt{P} \xrightarrow{d'} \mathtt{P''} \xrightarrow{d''} \mathtt{P'} \wedge d = d' + d''$.
- Time transitions are additive: $\mathtt{P} \xrightarrow{d} \mathtt{P'}$ and $\mathtt{P'} \xrightarrow{d'} \mathtt{P''}$ implies $\mathtt{P} \xrightarrow{d+d'} \mathtt{P''}$.
- Strong bisimulation ~ is a congruence.
- ET-LOTOS is upward compatible with LOTOS, according to the definition given in [NiS92], but for guarded specifications only.

References

[BLT 94b] T. Bolognesi, F. Lucidi, S. Trigila, *A Timed Full LOTOS with Time/Action Tree Semantics* in: T. Rus, C. Rathay, eds., Theories and Experiences for Real-Time System Development, Amast Series in Computing (Word Scientific, 1994), 205-237.

[ISO 8807] ISO/IEC-JTC1/SC21/WG1/FDT/C, *IPS - OSI - LOTOS, a Formal Description Technique Based on the Temporal Ordering of Observational Behaviour*, IS 8807, Feb. 1989.

[LéL 95a] L. Léonard, G. Leduc, *An Introduction to ET-LOTOS for the Description of Time-Sensitive Systems*, submitted for publication to: Computer Networks and ISDN Systems.

[LéL 95b] L. Léonard, G. Leduc, *A Formal Definition of Time in LOTOS*, internal report, Université de Liège, 1995.

[NiS 92] X. Nicollin, J. Sifakis, *An Overview and Synthesis on Timed Process Algebras,* in: K.G. Larsen, A. Skou, eds., Computer-Aided Verification, III (LNCS 575, Springer-Verlag, Berlin Heidelberg New York, 1992) 376-398. Also in: LNCS 600.

[QMF 94] J. Quemada, C. Miguel, D. de Frutos, L. Llana, *A Timed LOTOS Extension*, in: T. Rus, C. Rattray, eds., Theories and Experiences for Real-Time System Development, (World Scientific Pub., Inc 1994), 239-263.