Time Extended LOTOS

Source: Belgium[1] and Spain[2,3]

Draft: September 1995

1. Introduction

The Formal Description Technique LOTOS, defined in an International Standard [ISO 8807], is a method of defining the behaviour of an (information processing) system in a language with formal syntax and semantics. It has proven very successful in specifying many protocols and services.

However in LOTOS, nothing has been foreseen to handle the particular problem of describing time-dependent systems. Although possible in theory, a precise description of such systems in LOTOS is in most cases extremely tedious and results in extremely complex and poorly readable specifications. The need to formally specify time-dependent systems is real however. Most protocols are based on time-out mechanisms that are essential for the safety of their behaviour. Several new protocol mechanisms, as well as corresponding service facilities, strengthen this need. Isochronous data transfers, rate control, multimedia synchronization are some examples.

To remedy this problem, we introduce in the sequel a time extended version of LOTOS, called TE-LOTOS (for Time Extended LOTOS). It has been carefully designed to allow a clear and concise description of most time-dependent mechanisms, while remaining upward compatible[4] with existing LOTOS specifications.

TE-LOTOS is a super-set of LOTOS. All the LOTOS operators are kept unchanged. Simply, some of them are enriched with new features proper to the time environment. All these new features are optional however. A new delay operator is also introduced.

Following the method first introduced by Moler and Tofts in [MoT 90], the semantics of TE-LOTOS makes a clear distinction between timed and untimed aspects. The usual alphabet of actions of LOTOS is extended with a separate set, called the time domain. In TE-LOTOS, a process can evolve not only by accomplishing actions, but also by accomplishing time transitions. Intuitively, these time transitions describe the effects of the passing of time on the behaviour of a process. In the semantics, the set of axioms and inference rules defining the action transitions is completed with a separate set of axioms and rules defining the time transitions. Basically, the axioms and rules defining action transitions are the LOTOS one.

An alternative (but equivalent) definition of the semantics is also presented in Annex 2. In this second approach, which follows a method first introduced in [QuF 87], the usual alphabet of actions is transformed into a timed alphabet, by adding a time attribute to each untimed action. The time

---

[1] Contact: Luc Léonard and Guy Leduc, Institut Montefiore, Université de Liège, Belgium (edition of the main body).
[2] Contact: Carlos Miguel , Juan Quemada , Gualberto Rabay, Dpto. Ingeníera Telemática, ETSI Telecomunicación, Universidad Politécnica de Madrid, Spain (edition of Annex A)
[3] Contact: David de Frutos , Luis Llana, Dpto. Informática y Automática, Fac. Ciencias Matemáticas, Universidad Complutense, Madrid, Spain (edition of Annex A)
[4] A formal definition of the notion of upward compatibility is given in section 2.5.8

attribute associated with the occurrence of a timed action represents the passing of time from the preceding event. Therefore the occurrence of a timed transition is comparable to the occurrence of two transitions (a time transition followed by an action transition) in the first semantic approach.

The proposed model is based on the works by Leduc and Léonard [LeL 93, LéL 94, LéL 95] and by Quemada, Miguel et al [MFV 93, QMF 95].

2. Formal semantics and properties of TE-LOTOS

2.1. Data types and time domain

In TE-LOTOS, the data types are described in the Abstract Data Type language ACT ONE.

The time domain, denoted D, is defined as the set of values of a given data sort (D = Q(time)[1] where time is a LOTOS sort). 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 ∈ D such that: ∀ r ∈ D • 0≤r
- An element ∞ ∈ D such that: ∀ r ∈ D • r≤∞
- A commutative and associative operation "+ : D,D → D" such that:

  ∀ r,r1 ∈ D: r≤r1 ⇔ ∃ r' ∈ D • (r'+r1)=r

  ∀ r,r1 ∈ D: r+≤r+r1

  ∀ r ∈ D: r+0=r

  ∀ r ∈ D: r+∞=∞

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

  ∀ r,r1 ∈ D • r < r1 ⇔ (r ≤ r1 ∧ ¬ (r1 ≤ r))

  ∀ r,r1,r2 ∈ D • r1 ≤ r ⇒ (r - r1 = r2 ⇔ r1 + r2 = r)

  ∀ r,r1 ∈ D • r ≤ r1 ⇒ 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 TE-LOTOS in terms of Labelled Transition Systems (LTS), it must be countable, such as the rational numbers.

2.2 Notations

The following conventions are adopted in the sequel.

G denotes the countable set of observable gates. L = G ∪ {δ} denotes the alphabet of observable gates extended with δ, the special gate denoting successful termination (δ ∉ G). S denotes the set of sorts. V denotes the set of ground terms in the quotient term algebra associated with the ACT ONE specification (see section 2.4.1): $V = \bigcup_{s \in S} Q(s)$. CL = L × V^* denotes the set of observable actions. A = CL ∪ {i} denotes the alphabet of actions, where the symbol i is reserved for the unobservable internal action (i ∉ L). g (resp. a) denotes an element of G (resp. A): g ∈ G, cl ∈ CL, a ∈ A. $gv_1 \ldots v_n$ and $\delta v_1 \ldots v_n$ denote elements of CL, with the v_i's ∈ V. Capital Greek letters such as Γ will be used to

[1] Refer to 2.4.1 for the definition of this notation.

denote subsets of G. D denotes the countable time domain which is the alphabet of time actions. $D_\infty = D - \{\infty\}$, $D_{0\infty} = D - \{0,\infty\}$.

2.3 Syntax of the behaviour part of TE-LOTOS

The collection of TE-LOTOS behaviour expressions is defined by the following BNF expression where $g \in G$, t is a variable of sort time, $\Gamma \subseteq G$, T ranges over intervals of time values $t^-..t^+$, where $t^- \in D_\infty$, $t^+ \in D$, $d \in D_\infty$, $d1 \in D_{0\infty}$, $\tilde{X}$ represents a vector of process names, SP is a selection predicate (a Boolean expression or an equation), the e_i's represent either any s, with $s \in S$, or x, with $x \in V$, the o_i's represent either ?x:s, with x a variable of sort s, or !x, with $x \in V$, and the x_i's (resp. tx_i's) are variables (resp. terms) of sorts s_i's . The new features are printed in italics:

P ::= Q where $\tilde{X} := \tilde{Q}$[1]

Q ::= stop | exit(e_1,…e_n)*{T}* | $go_1…o_n$*{t in T}*[SP];Q | i*{t in T}*;Q | *Wait(d);Q* | Q[]Q | Q|[Γ]|Q | hide Γ in Q | Q>>accept x_1:s_1,…x_n:s_n in Q | Q[>Q | X | [SP]->Q | let x_1=tx_1,…x_n=tx_n in Q | choice x_1:s_1,…x_n:s_n[]Q

In $go_1…o_n${t in T}[SP];Q, {t in T} and [SP] are optional. In {t in T}, "t in" is optional, as well as "in T". If omitted, T = 0..∞ when g ≠ i and T = 0..0 when g = i. If omitted, [SP] = [true].

The binding powers of the operators are like in LOTOS. For the new operator, Wait(d) has the same power as action-prefix.

2.4 Semantics of TE-LOTOS

2.4.1 Mapping function on a LTS.

The mapping function between a TE-LOTOS specification and a (structured) Labelled Transition System (LTS) is rather complex. It involves several phases that we recall hereafter.

First phase : The flattening mapping

The purpose of the *flattening mapping* is to produce a *canonical TE-LOTOS specification*, *CLS* for short, where all identifiers are unique and defined at one global level. This function is partial since only static semantically correct specifications have a well-defined canonical form CLS.

CLS is a 2-tuple <CAS, CBS> composed of:

(i) a *canonical behaviour specification CBS*, i.e. a set of process definitions *PDEFS* with an initial process definition $pdef_0 \in PDEFS : CBS = <PDEFS, pdef_0>$.

A *process definition* is a pair consisting of a process variable *p* and a behaviour expression *B* : *pdef* = <*p*, *B*>.

(ii) a *canonical algebraic specification CAS*, i.e. an algebraic specification <*S, OP, E*> (*S* is a set of sorts, *OP* is a set of operations and *E* is the set of conditional equations defined on the signature <*S, OP*>) such that the signature <*S, OP*> contains all sorts and operations occurring in *CBS*.

Second phase : The derivation system of a data representation and the interpretation of CAS

[1] For convenience, we suppose, without lack of generality, that there is a single where-clause that gathers all the process declarations of the specification.

This phase consists of generating a *derivation system*, denoted *DS*, from the data representation *CAS* = <*S, OP, E*>. This derivation system is composed of axioms and inference rules generated by the conditional equations of *E*.

A congruence relation between ground terms (terms which do not contain variables) is induced by *CAS* : two ground terms t_1 and t_2 are called *congruent* w.r.t. *CAS*, simply denoted $t_1 = t_2$, iff

DS $\vdash t_1 = t_2$, i.e. it is possible to prove $t_1 = t_2$ from the axioms and the inference rules of the derivation system *DS*.

[t] denotes the set of all ground terms congruent to *t* w.r.t. *CAS*, i.e. intuitively *[t]* is the object represented by *t* or any of its equivalent representations.

The semantic interpretation of *CAS* = <*S, OP, E*> is the many-sorted algebra $Q(CAS) = <D_Q, O_Q>$, called the *quotient term algebra*, where

(i) D_Q is the set $\{Q(s) \mid s \in S\}$,

where $Q(s) = \{[t] \mid t$ is a ground term of sort $s\}$ for each $s \in S$; and

(ii) O_Q is the set of functions $\{Q(op) \mid op \in OP\}$,

where the $Q(op)$ are defined by $Q(op)([t_1], \ldots [t_n]) = [op(t_1, \ldots t_n)]$.

In this algebra, the terms with different representations but modelling the same object are collapsed.

Third phase : Mapping of CLS onto a LTS

The purpose of this last phase is the generation of a LTS. This generation is based on a transition derivation system.

The *transition derivation system* of a canonical TE-LOTOS specification CLS = <*CAS,CBS*> is composed of axioms and inference rules like those provided hereafter.

2.4.2 TE-LOTOS+

In section 2.3, we have defined the syntax of the behaviour part of TE-LOTOS.

The semantics of the generalized choice operator, given in section 2.4.4, is however defined in terms of an auxiliary operator, denoted `Achoice(d) x1:s1,…xn:sn[]P`, with `d` $\in$ D_∞ and with `choice x1:s1,…xn:sn[]P` = `Achoice(0) x1:s1,…xn:sn[]P`.

`Achoice(d) x1:s1,…xn:sn[]P` is not useful (when `d` $\neq$ `0`) to write a specification and therefore was not included in the syntax given section 2.3. However, after a time or an action transition a TE-LOTOS behaviour expression may turn into an expression where `Achoice(d) x1:s1,…xn:sn[]P` appears (with `d` $\neq$ `0`). The transition derivation system must then define the generation of a LTS for such behaviour expressions too.

To characterise these behaviour expressions including `Achoice(d) x1:s1,…xn:sn[]P`, we define a superset of TE-LOTOS, denoted TE-LOTOS+. The collection of TE-LOTOS+ behaviour expressions is defined by the following BNF expression, where `d1` $\in$ $D_{0\infty}$:

`P ::= Q` where $\tilde{X}$:= $\tilde{Q}$[1]

[1] For convenience, we suppose, without lack of generality, that there is a single where-clause that gathers all the process declarations of the specification.

Q ::= stop | exit(e_1,…e_n){T} | go$_1$…o$_n${t in T}[SP];Q | i{t in T};Q | Wait(d);Q | Q[]Q | Q|[Γ]|Q | hide Γ in Q | Q>>accept x_1:s_1,…x_n:s_n in Q | Q[>Q | X | [SP]->Q | let x_1=tx_1,…x_n=tx_n in Q | Achoice(d) x_1:s_1,…x_n:s_n[]Q

To summarize we can say that TE-LOTOS is the language that is used to write the specifications. TE-LOTOS is not "closed" however, in the sense that a TE-LOTOS process can evolve into a process that does not belong to TE-LOTOS anymore, because Achoice(d) x_1:s_1,…x_n:s_n[]Q appears in its syntax. But all the processes that can be derived from a TE-LOTOS behaviour expression belong to TE-LOTOS$^+$. Remark finally that not all the TE-LOTOS$^+$ behaviour expressions are a possible evolution of a TE-LOTOS behaviour expression. For the sake of simplicity however, we define the transition derivation system for any TE-LOTOS$^+$ behaviour expression, thus including the TE-LOTOS expressions.

2.4.3 Notations

P, P', Q, Q' denote TE-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 $\exists\, P' \bullet P \xrightarrow{d} P'$, 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.

2.4.4 Inference rules

In the following, the rules are presented in two columns. The rules concerned with action transitions are aligned on the left and the rules concerned with time transitions are aligned on the rigth.

The original LOTOS rules are left unchanged, except for action-prefix and exit which incorporate a new timed construct. The new rules appear in frames.

In the sequel, d, d1 $\in D_{0\infty}$, d', d$^-$ $\in D_\infty$, d$^+$ $\in D$, 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 (including time action), and therefore blocks the progression of time.

Inaction

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

Observable action-prefix

(AP1) $\text{go}_1\ldots\text{o}_n\{\text{t in 0..d}^+\}[\text{SP}];\text{P} \xrightarrow{gv_1\ldots v_n} [ty_1/y_1,\ldots ty_m/y_m, 0/t]P$

if DS $\vdash$ [ty1/y1,…tym/ym, 0/t]SP

where $v_i = [t_i]$ if $o_i = !t_i$

$v_i \in Q(s_i) = \{[t] \mid t \text{ is a ground term of sort } s_i\}$ if $o_i = ?x_i : s_i$

$\{y_1,\ldots y_m\} = \{x_i \mid o_i = ?x_i : s_i\}$

$[ty_j] = v_i$ if $y_j = x_i$ and $o_i = ?x_i : s_i$

$go_1 \ldots o_n\{t \text{ in } d^- + d \,..\, d^+ + d\}[SP];P \xrightarrow{d} go_1 \ldots o_n\{t \text{ in } d^- \,..\, d^+\}[[t+d/t]SP];[t+d/t]P$ (AP2)

$go_1 \ldots o_n\{t \text{ in } d^- \,..\, d^+\}[SP];P \xrightarrow{d} stop \quad (d > d^+)$ (AP3)

$go_1 \ldots o_n\{t \text{ in } d^- \,..\, d^+ + d\}[SP];P \xrightarrow{d} go_1 \ldots o_n\{t \text{ in } 0 \,..\, d^+\}[[t+d/t]SP];[t+d/t]P \quad (d > d^-)$ (AP4)

Remark the new construct added to the observable action-prefix: `{t in T}`. The `{T}` attribute restricts the time period during which an action can occur. In $go_1 \ldots o_n\{t \text{ in } d^- \,..\, d^+\}[SP];P$, the occurrence of an action at gate `g` is only possible after a delay of d^- time units and before a delay of d^+ time units. After d^+ time units, if no action has occurred at `g` yet, the process turns into `stop`.

In attributes of the form `{t in T}, t` is a variable of sort `time`. This variable is used to measure the delay an action was being offered when it occurred. It is thus instantiated when the action occurs. The `t` variable can appear in the selection predicate `SP`, if there is one.

Remark that if $\{t \text{ in } 0 \,..\, d^+\}$ is omitted, AP1 is equivalent to the LOTOS rule for observable action-prefix.

Internal action-prefix

(I1) $i\{t \text{ in } 0 \,..\, d^+\};P \xrightarrow{i} [0/t]P$

$i\{t \text{ in } d^- + d \,..\, d^+ + d\};P \xrightarrow{d} i\{t \text{ in } d^- \,..\, d^+\};[t+d/t]P$ (I2)

$i\{t \text{ in } d^- \,..\, d^+ + d\};P \xrightarrow{d} i\{t \text{ in } 0 \,..\, d^+\};[t+d/t]P \quad (d > d^-)$ (I3)

The same new construct as with the observable action-prefix is introduced.

There is no equivalent, for the internal action prefix, to rule (AP3). $i\{t \text{ in } d^- \,..\, d^+\};P$ cannot idle more than d^+ time units. If it reaches this limit, time is blocked. The only solution left is to accomplish `i`. This means that, in TE-LOTOS, the occurrence of `i` is compulsory. The semantics of `i{T};P` is that `i` *shall* occur during the interval of time defined by `T`[1]. On the contrary, due to rule (AP3), the semantics of $gd_1 \ldots d_n\{T\};P$ is that an action *may* occur at gate `g` (depending on the willingness of the environment), but only during the interval of time defined by `T`.

Here again, remark if $\{t \text{ in } 0 \,..\, d^+\}$ is omitted, I1 is equivalent to the LOTOS rule for internal action-prefix.

Delay prefixing

(D1) $\dfrac{P \xrightarrow{a} P'}{Wait(0);P \xrightarrow{a} P'}$

$Wait(d'+d);P \xrightarrow{d} Wait(d');P$ (D2)

$\dfrac{P \xrightarrow{d} P'}{Wait(d');P \xrightarrow{d+d'} P'}$ (D3)

`Wait(time);P` is a new operator: the delay prefixing. It expresses that `P` will be delayed by `time`.

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

Exit

(Ex1) $\texttt{exit}(e_1,\ldots e_n)\{0..d^+\} \xrightarrow{\delta v_1 \ldots v_n} \texttt{stop}$

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 $e_i = \text{any } s_i$

$$\texttt{exit}(e_1,\ldots e_n)\{d^-+d..d^++d\} \xrightarrow{d} \texttt{exit}(e_1,\ldots e_n)\{d^-..d^+\} \quad \text{(Ex2)}$$

$$\texttt{exit}(e_1,\ldots e_n)\{d^-..d^+\} \xrightarrow{d} \texttt{stop} \qquad (d > d^+) \quad \text{(Ex3)}$$

$$\texttt{exit}(e_1,\ldots e_n)\{0..d^++d\} \xrightarrow{d} \texttt{exit}(e_1,\ldots e_n)\{0..d^+\} \quad \text{(Ex4)}$$

The {T} attribute has the same meaning as with action prefixing. exit{T} can only perform δ during the time interval defined by T.

Choice

(Ch1) $\dfrac{P \xrightarrow{a} P'}{P[]Q \xrightarrow{a} P'}$

$$\frac{Q \xrightarrow{a} Q'}{P[]Q \xrightarrow{a} Q'} \quad \text{(Ch1')}$$

$$\frac{P \xrightarrow{d} P',\ Q \xrightarrow{d} Q'}{P[]Q \xrightarrow{d} P'[]Q'} \quad \text{(Ch2)}$$

Generalized choice

The semantics of choice $x_1{:}s_1,\ldots x_n{:}s_n[]P$ introduces an auxiliary operator, denoted Achoice(d) $x_1{:}s_1,\ldots x_n{:}s_n[]P$, where $d \in D_\infty$. By definition, choice $x_1{:}s_1,\ldots x_n{:}s_n[]P$ = Achoice(0) $x_1{:}s_1,\ldots x_n{:}s_n[]P$.

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

(GC2) $\dfrac{[tx_1/x_1, \ldots tx_n/x_n]P \xrightarrow{d'} P'',\ P'' \xrightarrow{a} P'}{\texttt{Achoice(d')}\ x_1{:}s_1, \ldots x_n{:}s_n[]P \xrightarrow{a} P'}$ if $d' > 0$

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

Remark that rule GC1 is the same as the LOTOS rule for the generalized choice if one replaces Achoice(0) $x_1{:}s_1,\ldots x_n{:}s_n[]P$ by choice $x_1{:}s_1,\ldots x_n{:}s_n[]P$. Rule GC2 concerns the actions accomplished by Achoice(d) $x_1{:}s_1,\ldots x_n{:}s_n[]P$, when d > 0. It is a new rule though relative to action transitions.

Parallel composition

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

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

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

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

Hide

(H1) $\dfrac{P \xrightarrow{a} P'}{\text{hide } \Gamma \text{ in } P \xrightarrow{a} \text{hide } \Gamma \text{ in } P'}$ ($a \notin \Gamma$)

(H2) $\dfrac{P \xrightarrow{a} P'}{\text{hide } \Gamma \text{ in } P \xrightarrow{i} \text{hide } \Gamma \text{ in } P'}$ ($a \in \Gamma$)

$$\frac{P \xrightarrow{d} P',\ \forall g \in \Gamma \bullet \left(P \overset{g}{\not\longrightarrow} \wedge\ \forall P''\forall d'<d \bullet \left(P \xrightarrow{d'} P'' \Rightarrow P'' \overset{g}{\not\longrightarrow}\right)\right)}{\text{hide } \Gamma \text{ in } P \xrightarrow{d} \text{hide } \Gamma \text{ in } P'} \quad \text{(H3)}$$

Rule (H3) expresses the *maximal progress* principle adopted in TE-LOTOS. This principle states that the hidden events must occur as soon as possible.

Enabling

(En1) $\dfrac{P \xrightarrow{a} P'}{P >> \text{accept } x_1{:}s_1, \ldots x_n{:}s_n \text{ in } Q \xrightarrow{a} P' >> \text{accept } x_1{:}s_1, \ldots x_n{:}s_n \text{ in } Q}$ (name{a} $\neq \delta$)

(En2) $\dfrac{P \xrightarrow{\delta v_1 \cdots v_n} P'}{P >> \text{accept } x_1{:}s_1, \ldots x_n{:}s_n \text{ in } Q \xrightarrow{i} [v_1/x_1, \ldots v_n/x_n]Q}$ $\forall i \bullet v_i \in Q(si)$

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

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) $\dfrac{P \xrightarrow{a} P'}{P [> Q \xrightarrow{a} P' [> Q}$ (name(a) $\neq \delta$)

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

(Di3) $\dfrac{P \xrightarrow{a} P'}{P [> Q \xrightarrow{a} P'}$ (name(a) = δ)

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

Guard

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

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

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

Let

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

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

Process instantiation

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

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

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

- The LOTOS rules are kept unchanged.
- The alphabet 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 D.
- There is no auxiliary function.

2.4.5 Time blockages

In TE-LOTOS, some processes block the progression of time. It means that for these processes, no time transition can be derived from the operational semantics. This situation is desired when the aim is to enforce the immediate occurrence of an internal event. This is the case for processes like `i{0..0};P` or `hide a in (a;stop)`. But it is possible in TE-LOTOS to define pathological processes where the time blockage is an undesired side-effect of the semantics. Examples are discussed in the following.

In TE-LOTOS, unguarded specifications[1] block the progression of time. For example: `P:= P` or `P:= a;Q[]P`.

[1] "P where $X_1 := P_1, \ldots X_n := P_n$" is a guarded Time Extended LOTOS specification if, by recursively substituting a finite number of times the expressions P_i's for the process identifiers X_i's occurring in P and in the P_i's themselves, it is

Other forms of time blockage can appear, but only with a dense time domain. For example, with a process like: `hide g in (g{t}[t>0];P)`. Such a process is in fact counter-intuitive in dense time. The maximal progress principle states that a hidden event must occur as soon as possible. However, if the time domain is dense, there is no earliest time at which `g` is enabled. Another case is the following: `choice t:time[] [t>0]-> Wait(t) i;P`. This process is also counter-intuitive. `i;P` stands for `i{0..0};P`, so `i` must occur immediately. However, it is impossible to determine the first occurrence of `i` that must occur in the choice context. More detailed explanations can be found in [LéL 95].

2.5. Properties

The proofs of these properties can be found in [LéL 95]

2.5.1. Consistency of the semantics

In [Gro 90], Groote explains that operational semantics with negative premises face the risk of being inconsistent, what results in the impossibility to derive a LTS for some behaviour expressions. On the other hand, the uniqueness of the LTS derived from a consistent semantics with negative premises is not certain. It is shown in [LéL 95] how one and only one LTS is derived for any behaviour expression in TE-LOTOS$^+$.

2.5.2. Time determinism

The time transitions are deterministic. This means that $\forall P \bullet (P \xrightarrow{d} P' \wedge P \xrightarrow{d} P'') \Rightarrow P' = P''$.

2.5.3. Time density

The time transitions are closed under the relation $\leq$:
$P \xrightarrow{d} \Rightarrow \forall\, d' \in]0,d] \bullet P \xrightarrow{d'}$.

Furthermore, $P \xrightarrow{d} P' \Rightarrow \forall\, d' \in]0,d[\bullet \exists\, d'' \bullet P \xrightarrow{d'} P'' \xrightarrow{d''} P' \wedge d = d' + d''$.

2.5.4. Additivity

The time transitions are additive: $P \xrightarrow{d} P'$ and $P' \xrightarrow{d'} P''$ implies $P \xrightarrow{d+d'} P''$.

2.5.5. Strong bisimulation

The definition of strong bisimulation in TE-LOTOS$^+$ follows.

Consider a LTS = $\langle S, A \cup D, T, s_0 \rangle$.
A relation $\underline{R} \subseteq S \times S$ is a strong bisimulation iff $\forall \langle B_1, B_2 \rangle \in \underline{R}$,$\forall\, \alpha \in A \cup D$, we have

(i) if $B_1 \xrightarrow{\alpha} B_1'$, then $\exists\, B_2'$ such that $B_2 \xrightarrow{\alpha} B_2'$ and $\langle B_1', B_2' \rangle \in \underline{R}$
(ii) if $B_2 \xrightarrow{\alpha} B_2'$, then $\exists\, B_1'$ such that $B_1 \xrightarrow{\alpha} B_1'$ and $\langle B_1', B_2' \rangle \in \underline{R}$

This is the classical definition of a strong bisimulation, where time transitions from `D` are considered as any other transitions. The strong bisimulation equivalence between two `LTS` is defined as follows.

---

possible to obtain an expanded Time Extended LOTOS specification "Q where $X_1 := Q_1, \ldots X_n := Q_n$" where Q and the Q_i's are guarded expressions, i.e. if all instantiations of Q_i's in X_j's are preceded by at least an action (observable or not) or a (non-zero) delay.

Definition

Two LTSs $Sys_1 = <S_1, A \cup D, T_1, s_{0_1}>$ and $Sys_2 = <S_2, A \cup D, T_2, s_{0_2}>$ are strong bisimulation equivalent, denoted $Sys_1 \sim Sys_2$, iff

$\exists$ a strong bisimulation relation $\underline{R} \subseteq S_1 \times S_2$, such that $<s_{0_1}, s_{0_2}> \in \underline{R}$

Property

In TE-LOTOS$^+$ strong bisimulation $\sim$ is a congruence.

This is very important in order to be able to replace a part of a TE-LOTOS$^+$ description by another strongly bisimilar process without changing the semantics of the description, i.e. the overall description remains strongly bisimilar to the original one.

Laws for strong bisimulation equivalence

All the laws listed in section B.2.2 (items a to k) of ISO 8807 (appendix B) are valid laws for strong bisimulation in TE-LOTOS.

New equivalence laws can be added.

Urgency

$i\{t\};P$	$\sim i;[0/t]P$	
$Wait(d);P \ [] \ i\{d^-..d^+\};Q$	$\sim i\{d^-..d^+\};Q$	if $d^+ < d$
$Wait(d);P \ [> \ i\{d^-..d^+\};Q$	$\sim i\{d^-..d^+\};Q$	if $d^+ < d$
$a\{d^-..d^+\};P \ [] \ i\{d_1^-..d_1^+\};Q$	$\sim a\{d^-..d_1^+\};P \ [] \ i\{d_1^-..d_1^+\};Q$	if $d^- \le d_1^+ \le d^+$
$a\{d^-..d^+\};P \ [] \ i\{d_1^-..d_1^+\};Q$	$\sim i\{d_1^-..d_1^+\};Q$	if $d_1^+ < d^-$
$a\{d^-..d^+\};P \ [> \ i\{d_1^-..d_1^+\};Q$	$\sim a\{d^-..d_1^+\};P \ [> \ i\{d_1^-..d_1^+\};Q$	if $d^- \le d_1^+ \le d^+$
$a\{d^-..d^+\};P \ [> \ i\{d_1^-..d_1^+\};Q$	$\sim i\{d_1^-..d_1^+\};Q$	if $d_1^+ < d^-$
$exit\{d^-..d^+\} \ [] \ i\{d_1^-..d_1^+\};Q$	$\sim exit\{d^-..d_1^+\} \ [] \ i\{d_1^-..d_1^+\};Q$	if $d^- \le d_1^+ \le d^+$
$exit\{d^-..d^+\} \ [] \ i\{d_1^-..d_1^+\};Q$	$\sim i\{d_1^-..d_1^+\};Q$	if $d_1^+ < d^-$

Time determinacy

$Wait(0);P$	$\sim P$	
$Wait(d);P \ [] \ Wait(d');Q$	$\sim Wait(d');(Wait(d-d');P \ [] \ Q)$	if $d' \le d$
$Wait(d);P \ \|[\Gamma]\| \ Wait(d');Q$	$\sim Wait(d');(Wait(d-d');P \ \|[\Gamma]\| \ Q)$	if $d' \le d$

Time additivity

$Wait(d); Wait(d');P$	$\sim Wait(d+d');P$
$Wait(d);stop$	$\sim stop$

Persistency

$a\{t\};P \ [] \ Wait(d);a\{t\};[t+d/t]P \sim a\{t\};P$

Others

$a\{d^-..d^+\};P \ [] \ a\{d_1^-..d_1^+\};P$	$\sim a\{min(d^-,d_1^-)..max(d^+,d_1^+)\};P$	if $max(d^-,d_1^-) \le min(d^+,d_1^+)$ and name(a) $\neq$ i
$exit\{d^-..d^+\} \ [] \ exit\{d_1^-..d_1^+\}$	$\sim exit\{min(d^-,d_1^-)..max(d^+,d_1^+)\};P$	if $max(d^-,d_1^-) \le min(d^+,d_1^+)$
$exit\{d^-..d^+\} \ \|[\Gamma]\| \ exit\{d_1^-..d_1^+\}$	$\sim exit\{max(d^-,d_1^-)..min(d^+,d_1^+)\}$	if $max(d^-,d_1^-) \le min(d^+,d_1^+)$
$a\{t \text{ in } d^-..d^+\}[SP];P$	$\sim stop$	if $\nexists \ t' \in [d^-,d^+] \bullet DS \vdash [t'/t] SP$ and name (a) $\neq$ i
$a\{t\}[d^- \le t \le d^+];P$	$\sim a\{t \text{ in } d^-..d^+\};P$	(name (a) $\neq$ i)

$a\{t\}[SP1];P \; [] \; a\{t\}[SP2];P \quad \sim a\{t\}[SP1 \vee SP2];P$

2.5.6 Expansion theorems

In this section, the behaviour expressions are in the following general format: $\sum_{i \in I} a_i\{t_i\}[SP_i(t_i)];\ P_i \; [] \; \sum_{j \in J} i\{t_j \text{ in } r_j^-..r_j^+\};\ P_j$. It is thus assumed that the elements of the summation (which means choice) can always be enumerated by some (possibly infinite) suitably chosen index sets I and J. The time interval is omitted with observable events because its action can be expressed with the selection predicate: $a\{t \text{ in } r^-..r^+\}[SP(t)];P \sim a\{t\}[SP(t) \wedge r^- \le t \le r^+];P$.

Let $P := \sum_{i \in I} a_i\{t_i\}[SP_i(t_i)];\ P_i \; [] \; \sum_{j \in J} i\{t_j \text{ in } r_j^-..r_j^+\};\ P_j$

and $Q := \sum_{k \in K} b_k\{t'_k\}[SP_k(t'_k)];\ Q_k \; [] \; \sum_{l \in L} i\{t'_l \text{ in } r'^-_l..r'^+_l\};\ Q_l$

We first introduce the rewriting function $Age(t,P)$ which is used to simplify the notations in the following theorems.

$$Age(t,P) = \sum_{i \in I} a_i\{t_i\}[SP_i(t_i+t)];\ [t_i+t/t_i]P_i \; [] \; \sum_{j \in J} i\{t_j \text{ in } r_j^- - t..r_j^+ - t\};\ [t_j+t/t_j]P_j$$

Remark that $Age(t,P)$ has no real meaning if $\exists\ j \in J \bullet t > r_j^+$. The definition in this case is thus purely conventional. This is not a problem, because in these expansion theorems, in all the circumstances where $Age(t,P)$ appears, such a situation cannot occur.

$$Wait(d);P \sim \sum_{i \in I} a_i\{t_i\}[SP_i(t_i-d) \wedge d \le t_i];\ [t_i-d/t_i]P_i \; [] \; \sum_{j \in J} i\{t_j \text{ in } r_j^-+d..r_j^++d\};\ [t_j-d/t_j]P_j$$

$$P|[\Gamma]|Q \sim \sum_{i \in I} \{a_i\{t_i\}[SP_i(t_i)];\ (P_i|[\Gamma]|Age(t_i,Q)) \mid name(a_i) \notin \Gamma \cup \{\delta\}\}$$
$$[] \sum_{j \in J} \{b_k\{t'_k\}[SP_k(t'_k)];\ (Age(t'_k,P)|[\Gamma]|Q_j) \mid name(b_k) \notin \Gamma \cup \{\delta\}\}$$
$$[] \sum \{c\{t_i\}[SP_i(t_i) \wedge SP_k(t_i)];\ (P_i|[\Gamma]|[t_i/t'_k]Q_k) \mid c=a_i=b_k,\ name(c) \in \Gamma \cup \{\delta\},\ i \in I,\ k \in K\}$$
$$[] \sum_{j \in J} \{i\{t_j \text{ in } r_j^-..r_j^+\};(P_j|[\Gamma]|Age(t_j,Q))\}$$
$$[] \sum_{l \in L} \{i\{t'_l \text{ in } r'^-_l..r'^+_l\};(Age(t'_l,P)|[\Gamma]|Q_l)\}$$

$$P \; [> \; Q \quad \sim Q \; [] \sum_{i \in I} \{a_i\{t_i\}[SP_i(t_i)];\ (P_i[>Age(t_i,Q))|name(a_i) \neq \delta\}$$
$$[] \sum_{j \in J} \{i\{t_j \text{ in } r_j^-..r_j^+\};\ (P_j[>Age(t_j,Q))$$
$$[] \sum_{i \in I} \{a_i\{t_i\}[SP_i(t_i)];\ P_i|name(a_i) = \delta\}$$

$$hide\ \Gamma\ in\ P \sim \sum_{i \in I} \{a_i\{t_i\}[SP_i(t_i)];\ hide\ \Gamma\ in\ P_i|name(a_i) \notin \Gamma\}$$

$$[] \sum_{i \in I} \{i\{r..r\};\text{hide } \Gamma \text{ in } [r/t_i]P_i \mid name(a_i) \in \Gamma,\ DS \vdash SP_i(r),\ i \in I\}$$

$$[] \sum_{j \in J} i\{t_j \text{ in } r_j^- .. r_j^+\};\ \text{hide } \Gamma \text{ in } P_j$$

Remark that in these expansion theorems, some branches which are generated are not necessary. The reason is that they can never occur due to the presence in competing branches of more urgent internal actions. On the other hand, their presence is harmless and avoiding them would increase too much the complexity of the expansion theorems.

2.5.7. Weak timed bisimulation

Let $d \in D$, $cl \in CL$ and ε the empty transition:

$P \stackrel{d}{\Rightarrow} Q$ iff $P \ (\stackrel{i}{\rightarrow})^* \ \stackrel{d1}{\rightarrow} \ (\stackrel{i}{\rightarrow})^* \ \stackrel{d2}{\rightarrow} \ (\stackrel{i}{\rightarrow})^* \ldots \stackrel{dn}{\rightarrow} (\stackrel{i}{\rightarrow})^* \ Q$ where $d = \sum_{i=1}^{n} d_i$

$P \stackrel{cl}{\Rightarrow} Q$ iff $P \ (\stackrel{i}{\rightarrow})^* \ \stackrel{cl}{\rightarrow} \ (\stackrel{i}{\rightarrow})^* \ Q$

$P \stackrel{\varepsilon}{\Rightarrow} Q$ iff $P \ (\stackrel{i}{\rightarrow})^* \ Q$

Consider a LTS = $<S, A \cup D, T, s_0>$.

A relation $\underline{R} \subseteq S \times S$ is a weak timed bisimulation iff $\forall <B_1, B_2> \in \underline{R}$,$\forall \alpha \in CL \cup D \cup \{\varepsilon\}$:

(i) if $B_1 \stackrel{\alpha}{\Rightarrow} B_1'$, then $\exists B_2'$ such that $B_2 \stackrel{\alpha}{\Rightarrow} B_2'$ and $<B_1', B_2'> \in \underline{R}$

(ii) if $B_2 \stackrel{\alpha}{\Rightarrow} B_2'$, then $\exists B_1'$ such that $B_1 \stackrel{\alpha}{\Rightarrow} B_1'$ and $<B_1', B_2'> \in \underline{R}$

Two LTSs $Sys_1 = <S_1, A \cup D, T_1, s_{0_1}>$ and $Sys_2 = <S_2, A \cup D, T_2, s_{0_2}>$ are weak timed bisimulation equivalent, denoted $Sys_1 \approx Sys_2$, iff

$\exists$ a weak timed bisimulation relation $\underline{R} \subseteq S_1 \times S_2$, such that $<s_{0_1}, s_{0_2}> \in \underline{R}$

$P \sim Q$ implies $P \approx Q$

Equivalence laws

`P ≈ i;P` (Remember that `i;P` is a shorthand notation for `i{0..0};P`)

`P [] i;P ≈ i;P`

`a;(P1 [] i;P2) [] a;P2 ≈ a;(P1 [] i;P2)`

$\approx$ is not a congruence in front of [] and [>, like the weak bisimulation in LOTOS, but also in other contexts like hiding. The following example, due to J.P. Courtiat, illustrates this:

`i{0..d};a;b{0..0};stop ≈ a;b{0..0};stop` but

`hide a in (i{0..d};a;b{0..0};stop) ≉ hide a in (a;b{0..0};stop).`

Due to the non-congruence in hiding contexts, the weakest congruence stronger than $\approx$ is still to be found.

2.5.8. Upward compatibility

Consider the LOTOS process algebra LOTOS = (OP, A, R_A^{OP}, ~) where OP is a set of operators, A is the alphabet of actions, R_A^{OP} is the set of operational semantics rules and ~ the strong bisimulation equivalence. Consider TE-LOTOS as the process algebra TE-LOTOS = (OP', A', $R_{A'}^{OP'}$, $\sim_E$) where OP' is a superset of OP, A' = A $\cup$ D is a superset of A, $R_{A'}^{OP'}$ is the new set of rules, and $\sim_E$ is the strong bisimulation equivalence in TE-LOTOS ($\sim_E$ denotes ~ as defined in section 2.5.5).

The definition retained for upward compatibility is the one given in [NiS92] where the following two requirements are stated. They are translated to the LOTOS framework as follows:

• Semantics conservation: $\forall$ r $\in R_A^{OP}$. r is valid in $R_A^{OP'}$ if it is applied on **LOTOS** terms.

The rules R_A^{OP} remain valid in TE-LOTOS as far as they are applied on LOTOS terms.

• Isomorphism: $\forall$ P, Q $\in$ **LOTOS** • P ~ Q iff P $\sim_E$ Q.

The theory of processes in LOTOS is isomorphic to that of the restriction of TE-LOTOS to constructs of LOTOS.

The semantics conservation is fulfilled but the isomorphism of the (TE-LOTOS, $\sim_E$) and the (LOTOS, ~) theories is only true for guarded specifications. With unguarded specifications the isomorphism between the theories is not true any more. For example, in LOTOS, P := stop and Q := Q are strong bisimulation equivalent, whereas in TE-LOTOS, P $\xrightarrow{d}$ but Q $\not\xrightarrow{d}$. Note that discriminating these two processes is considered more as an asset than as a shortcoming.

Moreover, for guarded specifications and as far as we have checked, the LOTOS laws for strong (resp. weak) bisimulation equivalence remain true $\forall$ P, Q $\in$ **TE-LOTOS** (i.e. not only on LOTOS terms), thereby preserving the LOTOS intuition in TE-LOTOS.

References

[Gro 90] J. F. Groote, *Transition system specifications with negative premises,* in: J.C.M. Baeten, J.W. Klop, eds., CONCUR '90, Theories of Concurrency: Unification and Extension, LNCS 458 (Springer - Verlag, Berlin, 1990) 332-341.

[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.

[LeL 93] G. Leduc, L. Léonard, *Comment rendre LOTOS apte à spécifier des systèmes temps réel?* , in: R. Dssouli, G. v. Bochmann, L. Lévesque, eds., Ingénierie des Protocoles - CFIP'93 (Hermès, Paris, 93).

[LéL 94] L. Léonard, G. Leduc, *An Enhanced Version of Timed LOTOS and its Application to a Case Study,* to appear in: R. Tenney, P. Amer, Ü. Uyar, eds., Formal Description Techniques, VI (North-Holland, Amsterdam, 1994).

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

[MFV 93] C. Miguel, A. Fernández, L. Vidaller, *Extending LOTOS towards performance evaluation*, in: M. Diaz, R. Groz, eds., Formal Description Techniques, V (North-Holland, Amsterdam, 1993) 103-118.

[MoT 90] F.Moller, C. Tofts, *A temporal calculus of communicating systems,* in: J.C.M. Baeten, J.W. Klop, eds., CONCUR '90, Theories of Concurrency: Unification and Extension, LNCS 458 (Springer - Verlag, Berlin Heidelberg New York, 1990) 401-415.

[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 95] J. Quemada, C. Miguel, D. de Frutos, L. Llana, *The Design of Timed Systems*, in: T. Rus, C. Rattray, eds., Theories and Experiences for Real-Time System Development, (World Scientific Pub., Inc 1995).

[QuF 87] J. Quemada, A. Fernandez, *Introduction of Quantitative Relative Time into LOTOS,* in: H. Rudin, C.H. West, eds., Protocol Specification, Testing and Verification, VII, (North-Holland, Amsterdam, 1987, ISBN 0-444-70293-8) 105-121.

Annex 1: Auxiliary functions

Although correctly defined, rules H3, En3 and GC3 are difficult to handle in practice. The problem is that their premises can be infinite: when the time domain is dense for H3 and En3 and when $Q(s_i)$ is infinite for some `i` for GC3.

It is possible however to take advantage of the "continuity" in the behaviour of the TE-LOTOS processes to reduce the complexity of these premises. It is clear for example, that with P:= a{3,5}; stop and $\Gamma = \{a\}$, the premise $(\forall\ d1<d \bullet \forall\ g \in \Gamma \bullet P_{d1} \stackrel{g}{\not\rightarrow})$ of rule H3 is simply equivalent to: $d \leq 3$.

On this basis, two auxiliary functions, denoted $NAB_\Gamma(t,P)$ and `Ci(t,P)`, are proposed.

$NAB_\Gamma(t,P)$[1] takes a time value `t` and a (closed) behaviour expression `P` as arguments and returns a Boolean. Its computation is based on the syntax of `P` and takes a maximal advantage of the information it contains. $NAB_\Gamma(t,P)$ verifies the proposition: $P \stackrel{t}{\rightarrow} \Rightarrow (NAB_\Gamma(t,P) \Leftrightarrow \forall\ g \in \Gamma \bullet \forall\ t' < t \bullet P_{t'} \stackrel{g}{\not\rightarrow})$. In other words, provided that $P \stackrel{t}{\rightarrow}$, $NAB_\Gamma(t,P)$ is equivalent to the second premise of rule H3. Note that the value of $NAB_\Gamma(t,P)$ has no meaning, and can even be undefined, if $P \stackrel{t}{\not\rightarrow}$. This is not a problem. As shown by rule (H3') below, the value of $NAB_\Gamma(t,P)$ only matters when $P \stackrel{t}{\rightarrow}$.

Thanks to this $NAB_\Gamma(t,P)$ function, rules H3 and En3 could be replaced by the following:

$$\text{(H3')}\quad \frac{P \stackrel{d}{\rightarrow} P',\ NAB_\Gamma(d,P)}{\texttt{hide}\ \Gamma\ \texttt{in}\ P \stackrel{d}{\rightarrow} \texttt{hide}\ \Gamma\ \texttt{in}\ P'}$$

$$\text{(En3')}\quad \frac{P \stackrel{d}{\rightarrow} P',\ \ NAB_{\{\delta\}}(d,P)}{P \gg Q \stackrel{d}{\rightarrow} P' \gg Q}$$

Similarly `Ci(t,P)`, which stands for "P Can Idle for t time units", takes a time value `t` and a (closed) behaviour expression `P` as arguments. It evaluates whether `P` *can idle* for `t` time units.

Rule GC3 can then be replaced by the following:

$$\text{(GC3')}\quad \frac{Ci(d,Achoice(d')\ x_1{:}s_1,\ldots x_n{:}s_n[]P)}{Achoice(d')\ x_1{:}s_1,\ \ldots\ x_n{:}s_n[]P \stackrel{d}{\rightarrow}\ Achoice(d+d')\ x_1{:}s_1,\ldots x_n{:}s_n[]P}$$

The definitions of $NAB_\Gamma(t,P)$ and `Ci(t,P)` are given in the following.

Definition of NAB

The definition of $NAB_\Gamma(tl,P)$ requires the definition of an auxiliary function: $APO_g(tl,P)$. $APO_g(tl,P)$, for All Possible Occurrences, builds a set of pairs. Each pair corresponds to a possible occurrence of `g` in `P`. The first element of a pair is the list of attributes associated with this occurrence of `g`. The second element is the selection predicate associated with it. In this selection predicate, the time variable is renamed τ and the variables defined in the attributes, ξ_i, where `i` is the rank of the corresponding attribute in the list.

[1] NAB is the acronym for No Action (from Γ are enabled in P) Before (time tl)

$$APOg(tl,stop) = \emptyset$$

$$APOg(tl,i\{t \text{ in } d^-..d^+\};P) = \emptyset$$

$$APOg(tl,exit(e_1\ldots e_n)\{t \text{ in } d^-..d^+\}) = \begin{cases} \emptyset & \text{if } g \neq \delta \\ \{e'_1\ldots e'_n, d^- \leq \tau \leq min(tl,d^+)\} & \text{if } g = \delta \end{cases}$$

where $e'_i = ?s$ if $e_i = any\ s$, and $e'_i = e_i$ otherwise

$$APOg(tl,ao_1\ldots o_n\{t \text{ in } d^-..d^+\}[SP];P) =$$

$$\begin{cases} \emptyset & \text{if } a \neq g \\ \{o'_1\ldots o'_n,\ [\xi_1/o_1,\ldots\xi_n/o_n,\tau/t]SP \wedge d^- \leq \tau \leq min(d^+,tl)\} & \text{if } a = g \end{cases}$$

where $o'_i = ?s$ if $o'_i = ?x{:}s$, and $o'_i = o_i$ otherwise

$$APOg(tl,Wait(d);P) = \begin{cases} [\tau - d/\tau]APOg(tl-d,P) & \text{if } tl > d \\ \emptyset & \text{if } tl \leq d \end{cases}$$

$$APOg(tl,P[]Q) = APOg(tl,P) \cup APOg(tl,Q)$$

$$APOg(tl,P|[\Gamma]|Q) = \begin{cases} APOg(tl,P) \cup APOg(tl,Q) & \text{if } g \notin \Gamma \cup \{\delta\} \\ Merge(APOg(tl,P),\ APOg(tl,Q)) & \text{if } g \in \Gamma \cup \{\delta\} \end{cases}$$

$$APOg(tl,P[>Q) = APOg(tl,P) \cup APOg(tl,Q)$$

$$APOg(tl,P>>accept\ x_1{:}s_1,\ldots x_n{:}s_n \text{ in } Q) = APOg(tl,P)$$

$$APOg(tl,hide\ \Gamma \text{ in } P) = \begin{cases} APOg(tl,P) & \text{if } g \notin \Gamma \\ \emptyset & \text{otherwise} \end{cases}$$

$$APOg(tl,[SP]->P) = \begin{cases} APOg(tl,P) & \text{if } SP \\ \emptyset & \text{otherwise} \end{cases}$$

$$APOg(tl,let\ x_1{=}tx_1,\ldots x_n{=}tx_n \text{ in } P) = APOg(tl,[tx_1/x_1,\ldots tx_n/x_n]\ P)$$

$$APOg(tl,Achoice(d)\ x_1{:}s_1,\ldots x_n{:}s_n[]P) = \bigcup_{[tx_i]\in Q(s_i)} [\tau + d/\tau]APOg(tl+d,[tx_1/x_1,\ldots\ tx_n/x_n]P)$$

$$APOg(tl,X_i \text{ where } X_1{:=}P_1,\ldots X_n{:=}P_n) = \begin{cases} APOg(tl,P_i) & \text{if } X_i \text{ where } X_1{:=}P_1,\ldots X_n{:=}P_n \text{ is a guarded spec} \\ \emptyset & \text{otherwise} \end{cases}$$

Merge takes two APO's as argument and returns an APO that is the set of all the pairs corresponding to a possible interactions between pairs from each argument.

$$Merge(APOg(tl,P),\ APOg(tl,\ Q)) =$$
$$\{(o_1\ldots o_n,SP) \mid (e_1\ldots e_n,SP1)\in APOg(tl,\ P) \wedge (f_1\ldots f_n,SP2)\in APOg(tl,Q) \wedge SP = (SP1 \wedge SP2)$$
$$\wedge \forall\ i = 1,\ldots n \bullet ((o_i = e_i = !v \wedge f_i = !w \wedge [v] = [w])$$
$$\vee\ (e_i = ?s \wedge o_i = f_i = !w \wedge w \in Q(s))$$
$$\vee\ (o_i = e_i = !v \wedge f_i = ?s \wedge v \in Q(s))$$
$$\vee\ (o_i = e_i = f_i = ?s))\}$$

Note that the value of APOg is counter-intuitive on unguarded specifications since they can possibly perform g. However it is not worth giving a correct value to APOg in this case (which would require a fixed point theory) because we do not consider APOg in those cases (see next definition, next proposition and rules H3' and En3').

$NAB_\Gamma(tl,P)$ is a Boolean expression. Intuitively, it is true if no action is possible on a gate in Γ before time tl.

$$NAB_\Gamma(tl,P) = \nexists\ t<tl \bullet \exists\ g\in\Gamma \bullet \exists\ (o_1\ldots o_n,SP)\in APOg(tl,P) \bullet \exists\ v_1\ldots v_n \bullet (\forall\ i = 1,\ldots n \bullet o_i=?s \Rightarrow v_i\in Q(s)) \wedge [v_1/\xi_1\ \ldots v_n/\xi_n,\ t/\tau]SP\})$$ [1]

Proposition

$$P \xrightarrow{tl} \Rightarrow NAB_\Gamma(tl,P) = ((P \not\xRightarrow{g} \forall\ g\in\Gamma) \wedge (\forall\ t<tl \bullet P \xrightarrow{t} P' \Rightarrow (P' \not\xRightarrow{g} \forall\ g\in\Gamma)))$$

Definition of Ci

$Ci(tm,stop) = true$

$Ci(tm,i\{t\ in\ d^-..d^+\};P) = tm \le d^+$

$Ci(tm,gd_1\ldots d_n\{t\ in\ d^-..d^+\}[SP];P) = true$

$Ci(tm,exit(d_1,\ldots d_n)\{t\ in\ d^-..d^+\}) = true$

$$Ci(tm,Wait(d);P) = \begin{cases} Ci(tm - d,P) & \text{if } tm>d \\ true & \text{otherwise} \end{cases}$$

$Ci(tm,P[]Q) = Ci(tm,P) \wedge Ci(tm,Q)$

$Ci(tm,P|[\Gamma]|Q) = Ci(tm,P) \wedge Ci(tm,Q)$

$Ci(tm,P[>Q) = Ci(tm,P) \wedge Ci(tm,Q)$

$Ci(tm,P>>accept\ x_1:s_1,\ldots\ x_n:s_n\ in\ Q) = Ci(tm,P) \wedge NAB_{\{\delta\}}(tm,P)$

$Ci(tm,hide\ \Gamma\ in\ P) = Ci(tm,P) \wedge NAB_\Gamma(tm,P)$

$$Ci(tm,[SP]->P) = \begin{cases} Ci(tm,P) & \text{if } SP \\ true & \text{otherwise} \end{cases}$$

$Ci(tm,let\ x_1=tx_1,\ldots\ x_n=tx_n\ in\ P) = Ci(tm,[tx_1/x_1,\ldots\ tx_n/x_n]P)$

$$Ci(tm,Achoice(d)\ x_1:s_1,\ldots\ x_n:s_n[]P) = \bigwedge_{[tx_i]\in Q(s_i)} Ci(tm+d,[tx_1/x_1,\ldots\ tx_n/x_n]P)$$

$$Ci(tm,X_i\ where\ X_1:=P_1,\ldots X_n:=P_n) = \begin{cases} Ci(tm,P_i) & \text{if } X_i \text{ where } X_1:=P_1,\ldots X_n:=P_n \text{ is a guarded spec} \\ false & \text{otherwise} \end{cases}$$

Proposition

$$Ci(d,P) \Leftrightarrow P \xrightarrow{d}$$

[1] Note that if $o_i \neq ?s$, v_i may be any value since there is no ξ_i variable in SP