A Calculus of Timed Refinement*

EXTENDED ABSTRACT

Kārlis Čerāns
Institute of Mathematics and Computer Science,
University of Latvia
E-mail: `karlis@mii.lu.lv`

1 Introduction

This paper presents CTR - a process algebraic framework for loose specification of time quantity sensitive operational behaviour of reactive systems.

The necessity and role of quantitative time information in reactive (open) system descriptions is nowadays quite appreciated[1], and there are already quite many specification formalisms, which alongside with system structural and/or behaviour event ordering specification permits to specify quantitative time information characterizing the valid behaviours (see e.g. [MF76, CC89, AD90, Wan91, GMMP89] as a few examples of such formalisms).

Quite naturally, many of the existing "timed" specification formalisms do allow in certain natural sense "loose" specifications of time information (it would be not that natural to determine at early system design stage exactly what are the time quantities between specific events going to be[2]), at least [MF76, AD90, Che92, Dan92, CGL93] to be mentioned.

The calculus CTR[3] presented in this paper is also designed to follow this tradition. Specifically, its aim is to provide a framework with mathematically precise and semantically justified notion of timed specification refinement (when one specification can be termed to be "more precise" than the other), concentrating on external behaviour specifications and branching time semantics. The model closest to CTR among those already present in the literature certainly is that of Timed Modal Specifications (TMS) introduced in [CGL93, CGL94]. We discuss in Section 5 in detail, what subset of CTR corresponds to a slight semantic reformulation[4] of TMS.

---

*The full version to be published as a technical report LU-IMCS-TRCS-94-??, Institute of Mathematics and Computer Science, University of Latvia, Riga 1994

[1] Even if the observer of a system is not directly concerned about the actual time quantities between externally observable events, it could be relevant for him/her to put some timing information in the specifications of the system components in order to ensure the correct *qualitative* ordering of the whole system output events.

[2] It may appear impossible even on the level of implementations due to various unpredictable execution environment fluctuations.

[3] A Calculus of Timed Refinement.

[4] Achieving, moreover, more intuitive properties of the specification refinement.

In essence, CTR is obtained by adding to the Milner's CCS [Mil89] the contruct of time interval prefix $[a,b].P$, where a,b are nonnegative real numbers, $0 \leq a \leq b$, possibly $b = \infty$. Such a specification is intended to describe all processes which first delay for some quantity $d \in [a,b]$, and then behave accordingly to some behaviour prescribed by P. (CTR can be seen as a generalization of Timed CCS due to Wang Yi in its version of [Wan90], where only delay prefixes of the form $[d,d].P$ were considered and denoted by $\epsilon(d).P$).

We define an operational semantics for CTR specification terms as a labelled transition system (each transition label denotes either some control action (externally visible or invisible), or it specifies some amount of time passage). Based on this semantics the relations of specification refinement $\trianglelefteq$ and equivalence $\dot{\approx}$ are defined in a bisimulation - like fashion[5].

The CTR specification equivalence and refinement relations are shown to obey the algebraic properties of equivalence/preorder and substitutivity (congruence/pre-congruence). On Timed CCS terms $\trianglelefteq$ induces a so-called divergence refinement relation, which is a slightly stronger property than the weak bisimulation $\approx$ of [Wan90]. $\dot{\approx}$ and $\trianglelefteq$ are also *decidable* for finite control specifications[6] (these include, for instance, networks of regular specifications).

Pragmatically, the CTR refinement relation $\trianglelefteq$ is based on the intuition of resolving underspecification of timing information. So, $[e,f].S \trianglelefteq [d,g].T$ whenever $d \leq e \leq f \leq g$ and $S \trianglelefteq T$ [7] (the tighter the delay interval is, the finer is the specification). On the extreme, the term $[0,\infty].S$ covers any specification of the form $[d,e].S$, together with the "empty" specification nil which never exhibits any observable behaviour but waiting. In fact, the term $[0,\infty].S$ would specify just the *possibility* of the system to exhibit the behaviour prescribed by S after some (possibly none) initial delay[8]. This is the only way of achieving in CTR looseness of "non-timed" transition behaviour of specifications.

On the other hand, the CTR refinement relation $\trianglelefteq$ admits abstraction from internal transitions of specifications and is focusing just on the *externally observable* behaviour. As an example, we have

$$(a.[d,e].\overline{b} \mid b.[f,g].\overline{c})\backslash b \trianglelefteq a.[d+f,e+g].\overline{c}.$$

Here on the right hand side we have a specification of all media which after recieving a message on the channel a wait for some time amount $\delta \in [d+f, e+g]$, after what a message on the channel c is delivered. On the left hand side we have decomposed such a general message transmission specification into more detailed one, where the transmission is done in two stages - first from the channel a to the internal channel b, then from b to c. For each stage the looseness of the delay specification is maintained. Observing the substitutivity property of $\trianglelefteq$, one can during further stages of the design make further refinements of

[5] As largest relations in the set of all specification terms, satisfying certain local "closeness" properties.

[6] To be precise, the decidability results hold for specifications whose initial delay bounds are given by integers (or at least rational numbers). In practice this is not a real restriction.

[7] This includes the case when $S = T$, due to the reflexivity of $\trianglelefteq$.

[8] In the notation of TMS the term $[0,\infty].a.S$ (observe the action label a necessary here) is written as $a_{\diamond}.S$ and is read semantically as "a transition a leading to S *may* be enabled".

the components responsible for both transmission stages[9] still ensuring that the resulting system[10] will be a refinement of the initial general specification $a.[d+f, e+g].\overline{c}$.

This example illustrates how CTR is conceptually suitable for formal top-down system development by stepwise refinement. One can start with a general and very loose specificaton which captures only a few essential features of the system being designed. The stepwise refinement process allows, in a formal manner, to incorporate into the specification additional design decisions, both of structural and timing nature, each refinement step possibly being done locally in some component of the system[11].

One should note, however, that there are a lot of different notions of specification refinement (or, specification/implementation relations, for that matter), using rather different mathematical models, one could look at, say, [BW94, Lam91, Dil89]. The refinement aspect inherent in CTR is that of time information strenghening in external communication ability specifications, and no claim is being made about any "superiority" of this refinement aspect to others.

In the rest of the paper, after introducing the CTR model in Section 2, in Section 3 we consider the definition and properties of the refinement relation. Sections 4 and 5 are devoted to study of some semantic variations of CTR.

2 CTR: Syntax and Semantics

We start with defining the syntax of CTR specification terms.

Let $\Lambda = \Delta \cup \overline{\Delta}$ with $\overline{\overline{\alpha}} = \alpha$ for all $\alpha \in \Lambda$ be a set of *external actions*, ranged over by α, β, and let τ be a distinct symbol representing the internal actions. $\mathcal{A}ct = \Lambda \cup \{\tau\}$ is a set of all actions, ranged over by a, b.

Let $\mathcal{V}ar$ be a finite set of process variables, ranged over by X. Then the set of *regular* CTR expressions over $\mathcal{V}ar$ is defined by the following grammar:

$$E ::= nil \mid X \mid [c,e].E \mid]c,e].E \mid a.E \mid E+E,$$

where $c \in \mathbf{R}^{+0}, e \in \mathbf{R}^{+0} \cup \{\infty\}, c \leq e$. Here besides the standard process algebraic constructs nil, action prefix $a.E$, choice $E+E$ and variable (see [Mil89]) we have introduced two kinds of delay interval prefix (left-closed and left-open). We let also $\epsilon(d).E$ to be another notation for $[d,d].E$.

We assume furthermore that for every $X \in \mathcal{V}ar$ we are given a regular CTR expression E_X (over $\mathcal{V}ar$) that contains variable occurrences only guarded either by action or some delay interval prefix[12]. We let $\mathcal{E} = \{X \stackrel{def}{=} E_X | X \in \mathcal{V}ar\}$ and call it a system of variable defining equations.

Given such a system $\mathcal{E}$ we call any regular CTR expresion over $\mathcal{V}ar$ a *regular CTR specification* over $\mathcal{V}ar$ and $\mathcal{E}$ (possibly not mentioning $\mathcal{V}ar$ and $\mathcal{E}$, if they are clear from the context).

[9] As a trivial possibility, one could tighten the message delivery delay intervals.

[10] Any system $(S|T)\backslash b$, where $S \trianglelefteq a.[d,e].\overline{b}$ and $T \trianglelefteq b.[f,g].\overline{c}$.

[11] Notice that the decidability of the refinement relation provides with (at least principal) possibility for automatic formal verification of the validity of every performed refinement step.

[12] In other words, all variable occurrences in E_X are allowed only within subterms of the form $a.E$, $[c,e].E$ or $]c,e].E$. Moreover, a delay prefix counts as a valid guard, only if $c > 0$.

Let $\mathcal{G}(\mathcal{V}ar, \mathcal{E})$ be a set of all regular CTR specifications over $\mathcal{V}ar$ and $\mathcal{E}$, ranged over by G. We define the set $\mathcal{S} = \mathcal{S}(\mathcal{V}ar, \mathcal{E})$ of CTR networks (CTR specifications), over $\mathcal{V}ar$ and $\mathcal{E}$, by the grammar:

$$S ::= G \mid S|S \mid S\backslash A \mid S[f],$$

where $A \subseteq \Delta$ and $f : \Delta \to \Delta$ is a bijection, extended to $\Lambda \to \Lambda$ by $f(\overline{\alpha})) = \overline{f(\alpha)}$ and $f(\tau) = \tau$. We let S and T to range over the set of CTR networks.

We define for CTR specifications an operational semantics. Following the design of a standard timed process calculi TCCS [Wan90], we base it on the transition relations $S \xrightarrow{a} T$, meaning that S can execute a control transition $a \in \mathcal{A}ct$ and afterwards exhibit a behaviour prescribed by T, and $S \xrightarrow{\epsilon(d)} T$ meaning that S can become T just by delaying for d time units ($d \in \mathbf{R}^{>0}$).

The new point is in defining the semantics for delay interval prefix. Our choice is to admit the ability of $[c, e].S$ to exhibit the behaviour prescribed by S after any $d \in [c, e]$ time units, the actual value of d to be chosen nondeterministically every time this construct is executed[13]. However, to facilitate the compositionality of the CTR framework, we are to define also what $[c, e].S$ is going to become (can become) after waiting for some smaller time amount (e.g., $d < c$). At least three possible variants initially come into mind, depending on actually *when* during the delay the nondeterminism is being resolved. These variants correspond to (i) choosing in the beginning (early choice), (ii) choosing gradually during the execution (inline choice), and (iii) choosing when to leave (late choice). We shall work out in most detail (iii). (ii) is considered in Section 4 (showing also its relation to the late choice semantics), while (i) can be argued to be not suitable as a basis of timed refinement framework[14].

So, the execution of $[c, e].S$ in time begins with a deterministic and synchronous decreasing of both its higher and lower time bounds: $[c, e].S \xrightarrow{\epsilon(d)} [c-d, e-d].S$ for $d \le c$. When $[0, e].S$ is reached, the choice comes in. We still let $[0, e].S \xrightarrow{\epsilon(d)} [0, e-d].S$ for $d \le e$, however, now we want the specification to be able to choose to exhibit the behaviour prescribed by S immedilately. The design decision taken in CTR is to introduce a special new kind of internal transition $\xrightarrow{i}$ whose effect is to "leave the delay prefix": $[0, e].S \xrightarrow{i} S$. This construction materializes the principle of letting the specification itself to make all time nondeterministic choice without any influence from the environment.

Formally, we require $i \notin \mathcal{A}ct$ and let $\mathcal{A}ct^+ = \mathcal{A}ct \cup \{i\}$ to be ranged over by a^+. The whole operational semantics of CTR is given in Table 1 and Table 2 (we assume $f(i) = i$ for any relabelling function f). $x \dot{-} y$ means $max\{x - y, 0\}$ and $[\!($ stands for any kind of left end of the interval (or [). In the delay rules for $\xrightarrow{\epsilon(d)}$ we require $d \in \mathbf{R}^{>0}$. ν stands for either a^+, or $\epsilon(d)$.

For parallel composition, the sorts $Sort(d, S) \subseteq \Lambda$ in the side condition of the delay rule are computed in a standard way, as follows:

[13] This would amount to having $[c, e].S \xrightarrow{\epsilon(d)} S$ for all $d \in [c, e]$. As we shall see soon, a proper description of this property in CTR would be $[c, e].S \xrightarrow{\epsilon(d)}_I S$.

[14] In a sense, it requires the choice in the model to be done before the choice is performed in the real modelled system. We discuss this issue more precisely in Section 3.

$$
\frac{--}{nil \xrightarrow{\epsilon(d)} nil} \qquad \frac{--}{a.S \xrightarrow{a} S} \qquad \frac{--}{\alpha.S \xrightarrow{\epsilon(d)} \alpha.S} \qquad \frac{S \xrightarrow{\nu} S'}{X \xrightarrow{\nu} S'}[X \stackrel{def}{=} S] \in \mathcal{E}
$$

$$
\frac{S \xrightarrow{a} S'}{S+T \xrightarrow{a} S'} \qquad \frac{T \xrightarrow{a} T'}{S+T \xrightarrow{a} T'} \qquad \frac{S \xrightarrow{i} S'}{S+T \xrightarrow{i} S'+T} \qquad \frac{T \xrightarrow{i} T'}{S+T \xrightarrow{i} S+T'}
$$

$$
\frac{S \xrightarrow{\epsilon(d)} S' \;\; T \xrightarrow{\epsilon(d)} T'}{S+T \xrightarrow{\epsilon(d)} S'+T'} \qquad \frac{S \xrightarrow{a^+} S'}{S|T \xrightarrow{a^+} S'|T} \qquad \frac{T \xrightarrow{a^+} T'}{S|T \xrightarrow{a^+} S|T'} \qquad \frac{S \xrightarrow{\alpha} S' \;\; T \xrightarrow{\overline{\alpha}} T'}{S|T \xrightarrow{\tau} S'|T'}
$$

$$
\frac{S \xrightarrow{\epsilon(d)} S' \;\; T \xrightarrow{\epsilon(d)} T'}{S|T \xrightarrow{\epsilon(d)} S'|T'}[Sort(d,S) \cap \overline{Sort(d,T)} = \emptyset] \qquad \frac{S \xrightarrow{a^+} S'}{S\backslash A \xrightarrow{a^+} S'\backslash A}[a, \overline{a} \notin A]
$$

$$
\frac{S \xrightarrow{\epsilon(d)} S'}{S\backslash A \xrightarrow{\epsilon(d)} S'\backslash A} \qquad \frac{S \xrightarrow{a^+} S'}{S[f] \xrightarrow{f(a^+)} S'[f]} \qquad \frac{S \xrightarrow{\epsilon(d)} S'}{S[f] \xrightarrow{\epsilon(d)} S'[f]}
$$

Table 1: Transition rules for standard process algebraic constructions

$$
\frac{-- \quad [d \leq e]}{[c,e].S \xrightarrow{\epsilon(d)} [c \dot{-} d, e-d].S} \qquad \frac{--}{[0,e].S \xrightarrow{i} S} \qquad \frac{S \xrightarrow{a^+} S'}{[0,0].S \xrightarrow{a^+} S'} \qquad \frac{S \xrightarrow{\epsilon(d)} S'}{]c,e].S \xrightarrow{\epsilon(d+e)} S'}
$$

$$
\frac{-- \quad [d \leq c]}{]c,e].S \xrightarrow{\epsilon(d)}]c-d, e-d].S} \qquad \frac{-- \quad [c < d \leq e]}{]c,e].S \xrightarrow{\epsilon(d)} [0, e-d].S}
$$

Table 2: Transition rules for delay prefix

- $Sort(d, \alpha.S) = \{\alpha\}, \qquad Sort(d, \tau.S) = Sort(d, nil) = \emptyset$,
- $Sort(d, [c,e].S) = Sort(d \dot{-} e, S)$,
- $Sort(d, X) = Sort(d, S)$, where $[X \stackrel{def}{=} S] \in \mathcal{E}$
- $Sort(d, S+T) = Sort(d, S|T) = Sort(d,S) \cup Sort(d,T)$
- $Sort(d, S[f]) = f(Sort(d,S)$ and $Sort(d, S\backslash A) = Sort(d, A) \setminus (A \cup \overline{A})$.

Following the design of Timed CCS, we have incorporated in our framework the principle of *maximal progress*, saying that an enabled internal τ transition has to be executed before the time progresses further[15].

In the semantics one should observe that unlike the $\xrightarrow{\tau}$ transitions, $\xrightarrow{i}$ does not resolve the nondeterministic choice $S+T$ (as do not the delay transitions). It is to be observed also that a specification of the form $[0,0].S$ does not need

[15]There are no delay transitions for the $\tau.S$ terms, also the above side condition with sorts in parallel composition delay rule is just for this reason - a time can pass only until the point where a synchronization between componets is enabled. Note that this does not apply to $\xrightarrow{i}$.

to have any $\xrightarrow{i}$ transitions in order to "leave" the delay prefix. Intuitively this reflects the idea that $\xrightarrow{i}$ corresponds to an active choice, but in case of $[0,0].S$ no (timed) choice has been left.

Using the notation $\epsilon(d).S$ for $[d,d].S$ we can interpret all Timed CCS (network) terms of [Wan90] as CTR terms. It can be seen that this embedding preserves also the TCCS operational semantics (modulo ignoring $\xrightarrow{i}$ transitions which for TCCS terms can be essentially only of the form $S \xrightarrow{i} S$).

3 Refinement and Equivalence

In this section the CTR specification refinement and equivalence relations are defined on the basis of their operational behaviour. We recall also that the refinement relation is going to embody the idea of resolving timed underspecification (what operationally appears as timed non-determinism).

For S being a refinement of T, $S \trianglelefteq T$, we first require that any transition sequence (behaviour) of S is to be matched by a corresponding transition sequence of T[16], the resulting states of these sequences also being in the relation $\trianglelefteq$. In a sense, a more general specification needs to admit more behaviours.

However, our notion of the refinement is not just that of simulation. We require also that certain behaviours of T, namely those not involving $\xrightarrow{i}$ transitions, and not involving also delays, should be matched appropriately on the side of S. Since the $\xrightarrow{i}$ and delay transitions are the only ones possibly resolving the timed nondeterminism ($\xrightarrow{i}$ is doing that in an "active" way, while some delay transitions do that in a "passive" way), this requirement means that all behaviours *not* resolving timed nondeterminism are to be maintained (modulo the abstraction from $\xrightarrow{\tau}$) during the refinement process.

Formally, let $S \xrightarrow{a}_\circ S'$ denote $S(\xrightarrow{i})^* \xrightarrow{a} (\xrightarrow{i})^* S'$, $S \xRightarrow{\varepsilon} S'$ denote $S \xrightarrow{\tau}^* S'$ and $S \xRightarrow{\alpha} S'$ stand for $S \xRightarrow{\varepsilon}\xrightarrow{\alpha}\xRightarrow{\varepsilon} S'$. Moreover, let $\xRightarrow{\varepsilon}_\circ = (\xrightarrow{\tau} \cup \xrightarrow{i})^*$ and $S \xRightarrow{\alpha}_\circ S'$ mean $S \xRightarrow{\varepsilon}_\circ \xrightarrow{\alpha} \xRightarrow{\varepsilon}_\circ S'$.

In what follows, we prepare for defining the matching of delay transitions. For $S \in \mathcal{S}$ and $d \in \mathbf{R}^{>0}$ we call any finite sequence $\langle S_0, d_0\rangle, \langle S_1, d_1\rangle, \ldots, \langle S_n, d_n\rangle$ such that $S \xRightarrow{\varepsilon}_\circ S_0$, $d_0 = 0$, $d_n = d$ and for every $i = 1, \ldots, n-1$ we have $S_i \xrightarrow{\epsilon(d_{i+1}-d_i)} S_i' \xRightarrow{\varepsilon}_\circ S_{i+1}$ a *base sequence* for an $\langle S, d\rangle$-trace. The $\langle S, d\rangle$-*trace* itself associated with this base sequence is defined to be the function $f : [0,d] \to \mathcal{S}$ such that $f(d_i + \delta) = S_i \xrightarrow{\epsilon(\delta)}$, where $i = 1, \ldots, n$ and $\delta \in [0, d_{i+1} - d_i[$ [17]. We denote by $S \xRightarrow{\epsilon(d)}_\circ$ the set of all $\langle S, d\rangle$-traces.

For $f \in S \xRightarrow{\epsilon(d)}_\circ$ and $g \in T \xRightarrow{\epsilon(d)}_\circ$, given a relation $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S}$, we say that $\langle f, g\rangle \in \mathcal{R}^t$ if and only if $\langle f(d'), g(d')\rangle \in \mathcal{R}$ for every $d' \in [0,d]$.

For $S \in \mathcal{S}$ we let $S \xrightarrow{\tau^\infty}_\circ$ to denote the fact of existence of infinite chain of $\xrightarrow{\tau}_\circ$ transitions starting at S: $S \xrightarrow{\tau}_\circ S_1 \xrightarrow{\tau}_\circ S_2 \xrightarrow{\tau}_\circ \cdots$.

[16]This matching is abstracting from the externally non-observable transitions $\xrightarrow{i}$ and $\xrightarrow{\tau}$.

[17]By $S \xrightarrow{\epsilon(\delta)}$ we have denoted the unique T such that $S \xrightarrow{\epsilon(\delta)} T$. In Section 4 it will be demonstrated, how a trace can be defined also in the absence of this unicity.

Let μ range over $\Lambda \cup \{\varepsilon\}$ (μ is either α, or ε).

Definition 3.1 *A relation $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S}$ is a refinement relation, if for any $\langle S, T\rangle \in \mathcal{R}$*

1. $S \stackrel{\mu}{\Longrightarrow}_\circ S'$ *implies* $\exists T'. T \stackrel{\mu}{\Longrightarrow}_\circ T'$ *so that* $\langle S', T'\rangle \in \mathcal{R}$;
2. $T \stackrel{\mu}{\Longrightarrow} T'$ *implies* $\exists S'. S \stackrel{\mu}{\Longrightarrow} S'$ *so that* $\langle S', T'\rangle \in \mathcal{R}$;
3. $f \in S \stackrel{\epsilon(d)}{\Longrightarrow}_\circ$ *implies existence of* $g \in T \stackrel{\epsilon(d)}{\Longrightarrow}_\circ$ *such that* $\langle f, g\rangle \in \mathcal{R}^t$;
4. $S \stackrel{\tau^\infty}{\longrightarrow}_\circ$ *implies* $T \stackrel{\tau^\infty}{\longrightarrow}_\circ$.

We say that S is a refinement of T, and write $S \trianglelefteq T$ whenever there exists a refinement relation $\mathcal{R}$ containing $\langle S, T\rangle$ (In other words, $\trianglelefteq$ is the largest refinement relation).

Observational equivalence is a refinement whose inverse is also a refinement[18].

A few further comments on the given definition are in order. The intuitive explanation of the refinement given in the beginning of the section motivates only its first three requirements (observing the specific way of defining a matching for timed transitions). Without including the fourth clause in the definition we would have a so-called *simple* refinement relation, which we denote by $\triangleleft$. It can be shown that $\triangleleft$ obeys the essential mathematical properties of transitivity and substitutivity. However, this relation is in a sense too large since $S \triangleleft T$ holds also in some cases, when it is semantically undesirable, for instance, when $T = a.nil$ and $S \stackrel{def}{=} a.nil + \tau.S$ [19].

The fourth clause in the refinement definition is pragmatically meant to forbid such examples (at the same time not forbidding too much). Though this clause is not of the form of "matching requirement", traditional for process algebraic equivalence and preorder relations, it is hoped that it would pragmatically fit well with the system refinement process.

To facilitate that we suggest a pragmatic requirement that a design in CTR should not admit "instantaneous" loops of internal transitions[20]. As loops of internal transitions are usually essential in the process algebraic system design, this requirement would mean just that any such loop possibly occurring in

[18] In this paper we do not further explore the notion of specification equivalence. However, it is clearly relevant for the theory.

[19] One most likely would not want the "time-blocking" specification S to be a refinement of a perfectly well defined specification willing to synchronize with its environment on a at the moment when the environment is ready to do so (and not necessarily "immedilately"). The simple refinement relation would be satisfactory, however, if one managed to exclude the possibilities of writing "time-blocking" specifications at all. This does not seem feasible while following the process algebraic specification style, and sticking to the maximal progress principle.

[20] In a modelling methodology, where we do take into account the actual time quantities, it might be admitted that some bounded amount of transitions are performed "instantaneously" (if that helps to simplify the model). However, for unbounded chains or loops of internal actions, it is no longer realistic that they could be executed within 0 units of time.

the specification should have its delay being strictly greater than 0 (a $]0,d].S$ construct executed in every round of the loop would be sufficient) [21].

Yet another variation of the refinement notion could be obtained by extending the set of "necessary" behaviours which, when present in a specification, should be matched appropriately also on the side of implementation[22]. A natural such extension is obtained by observing that *some* delay transitions of CTR specifications, namely those of $[c,e].S$ waiting for $d \leq c$ time units, are not resolving any timed non-determinism, and are in one or other form to be present also in all refinements of this specification.

However, a refinement relation which in case of T refined by S would require all "necessary" delay transitions of T to be matched by similar transitions of S (also done in the necessary mode, to ensure the transitivity of the refinement) appears to be too strong. To see that, look at the valid CTR refinement

$$[1,2].\tau.[1,2].\overline{c}.nil \trianglelefteq [2,4].\overline{c}.nil,$$

what is just a transformed and instantiated version of the transmission media refinement example considered in the introduction[23]. One can see that the right hand side is to delay for 2 time units in the "necessary mode". However, while it is clear that the left hand side also will necessarily delay for at least 2 time units before offering $\overline{c}$, there is no single chain of delay transitions of length 2 (possibly interleaved with $\xrightarrow{\tau}$) which could also be performed in the necessary mode, without any active or passive resolving of the time nondeterminism[24]. Thus the requirement of simple matching the "necessary" delay transitions would invalidate the above refinement instance, what could seriosly question the pragmatical applicability of the whole refinement relation[25] [26].

---

[21] As can be clearly seen from the definition of $\trianglelefteq$, this requirement has not been formally incorporated into the definition. All what has been said is, if the implementation (the stronger specification) admits such a loop for one or other reason, then also the more general specification has to have it admitted (it could be done as, for instance, in $S \stackrel{def}{=} a.nil + \tau.S$). Sometimes indeed some "time-blocking" effects could be useful to say the fact that a certain action should be immediate (ideally, of course, one should have other direct means of expressing that), however, that has nothing in common with the situation when a real chain (loop) of internal transitions is to be executed.

[22] For $\trianglelefteq$ these are just all behaviours not containing $\xrightarrow{i}$, and not containing delays.

[23] $(a.[d,e].\overline{b} \mid b.[f,g].\overline{c})\backslash b \trianglelefteq a.[d+f, e+g].\overline{c}$ with $d = f = 1$ and $e = g = 2$, here we consider the situation where the initial a has already been performed.

[24] Observe that already after 1 time unit of waiting the further performed delay of $[1,2]$ appears to be only in the possible mode due to the other existing alternative to take $\xrightarrow{i}$ at that moment.

[25] The framework of TMS in [CGL93, CGL94] is based on this idea of requiring the matching of "necessary" delay transitions in the direction from the specification to the implementation. Therefore in that framework only a reduced version of the considered example was possible (one does get a valid refinement instance when either $d = e$, or $f = g$, that is, when one of the components is deterministic).

[26] The same example $[1,2].\tau.[1,2].\overline{c}.nil \trianglelefteq [2,4].\overline{c}.nil$ can be used also to invalidate the "early choice" semantics for CTR, as discussed in Section 2 (provided we keep our schema of defining the refinement).

Indeed, as we want a delay transition of the implementation, say $\xrightarrow{\epsilon(0.1)}$, leading to $\epsilon(1.3).\tau.[1,2].\overline{c}.nil$, to be matched on the side of the specification, we need to determine at

We turn to the mathematical properties of $\trianglelefteq$. Due to the space constraints their proofs are mainly postponed to the full version.

Fact 3.2 *The relation $\trianglelefteq$ is reflexive and transitive.*

Lemma 3.3 *$\trianglelefteq$ is preserved by CTR parallel composition, restriction and relabelling operations.*

Proof: We prove that for any refinement relation $\mathcal{R}$ refinement relations are also $\mathcal{R}^+ = \{\langle S|T, P|T\rangle | \langle S, P\rangle \in \mathcal{R}\}$, $\mathcal{R}^+_{rs} = \{\langle S\backslash A, P\backslash A\rangle | \langle S, P\rangle \in \mathcal{R}\}$ and $\mathcal{R}^+_{rl} = \{\langle S[f], P[f]\rangle | \langle S, P\rangle \in \mathcal{R}\}$. The proof is a rather straightforward check, including the case of timed matching of parallel composition. $\square$

Lemma 3.4 *If $T \xrightarrow{i} S$, then $S \trianglelefteq T$.*

Proof: We show that the relation $\{\langle S, T\rangle | S = T \text{ or } T \xrightarrow{i} S\}$, is a refinement. The proof is a matter of combining definitions, given the following fact.

Fact 3.5 *If $T \xrightarrow{i} S$ and $T \xrightarrow{a} T'$, then either $S \xrightarrow{a} T'$, or there exists an S' such that $S \xrightarrow{a} S'$ and $T' \xrightarrow{i} S'$.*

Proof: By analyzing, whether $T \xrightarrow{i} S$ and $T \xrightarrow{a} T'$ are performed in distinct regular components of T, or in the same. $\square$

In what follows we show what relation the CTR refinement $\trianglelefteq$ induces in the set of time determinstic ("fully specified") process algebraic terms (those of Timed CCS). Let $\mathcal{P}$ denote the set of all TCCS (network) processes ($\mathcal{P} \subseteq \mathcal{S}$).

Following [Wan90], a relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a (weak) bisimulation, if for all $\nu \in \mathcal{A}ct \cup \{\epsilon(d) | d \in \mathbf{R}^{>0}\}$ and all $\langle P, Q\rangle \in \mathcal{R}$

- whenever $P \stackrel{\nu}{\Longrightarrow} P'$ then $\exists Q'.Q \stackrel{\nu}{\Longrightarrow} Q'$ so that $\langle P', Q'\rangle \in \mathcal{R}$ and
- whenever $Q \stackrel{\nu}{\Longrightarrow} Q'$ then $\exists P'.P \stackrel{\nu}{\Longrightarrow} P'$ so that $\langle P', Q'\rangle \in \mathcal{R}$.

Here $X \stackrel{\epsilon(d)}{\Longrightarrow} Y$ is defined to mean $X(\xrightarrow{\tau})^* \xrightarrow{\epsilon(d_1)} (\xrightarrow{\tau})^* \cdots \xrightarrow{\epsilon(d_k)} (\xrightarrow{\tau})^* Y$, where $d_1 + \cdots + d_k = d$. The largest TCCS weak bisimulation is denoted by $\approx$.

Fact 3.6 *For P and Q being TCCS processes, $P \trianglelefteq Q$ implies $P \approx Q$.*

To characterize the relation induced by $\trianglelefteq$ on TCCS terms precisely, let a relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ be called a *divergence refinement* iff

- $\mathcal{R}$ is a weak bisimulation, and
- for any $\langle P, Q\rangle \in \mathcal{R}$ whenever $P \xrightarrow{\tau^\infty}$, then also $Q \xrightarrow{\tau^\infty}$.

Let $\triangleleft_D$ be the largest divergence refinement for TCCS processes.

Theorem 1 *For P and Q being TCCS processes, $P \trianglelefteq Q$ if and only if $P \triangleleft_D Q$.*

---

this very moment, how long the actual delay of $[2, 4]$ is going to be. Doing that, we clearly can not keep the further flexibility inherent in the second implementation's $[1, 2]$ construct.

To prove the theorem we first give an equivalent characterization of $\approx$ and $\lhd_D$ using the form of "continuous matching" for delays[27]. Later on, a determinacy of $\xrightarrow{\epsilon(d)}$ transitions, together with the maximal progress and a few other CTR transition system properties are exploited.

4 Inline Choice Semantics

This section concerns defining alternative semantics for CTR terms following the idea of inline, or ongoing choice, where the "speed" of execution at every moment is contributing to when a specification is going to leave its delay interval prefix. Surprisingly enough, a refinement defined naturally on the basis of this really wildly nondeterministic transition system, coincides with the "standard" CTR refinement $\unlhd$. For the sake of simplicity of technical treatment we consider here only specifications in reduced CTR, where we disallow left-open delay interval prefixes $]c, e]$.

First we define the inline choice operational semantics as the transition relation $\longrightarrow_I$. For that we can take all the rules defining the "standard" semantics $\longrightarrow$ in Table 1 and Table 2 (with changing syntactically $\longrightarrow$ to $\longrightarrow_I$) except for two. First of those to be changed is the main rule for delay of the delay prefix, which is to read now as

$$[c,e].P \xrightarrow{\epsilon(d)} [c',e'].P \text{ whenever } max(0, c-d) \leq c' \leq e' \leq e-d.$$

Intuitively this rule means that during the delay of d time units the lower delay bound is reduced by no more than d, and the higher bound is reduced by not less than d. Any choice in between can be taken during the runtime.

To be able to define adequately the semantics of the delay rule of the parallel composition (we can not work just with the $Sort$ function, as in case of $\longrightarrow$), we have to refine a bit the whole definition of the relation $S \xrightarrow{\epsilon(d)}_I T$ by basing it not solely on the fact of *existence* of such transition, but rather on the explicit exhibition of the whole trace leading in d time units from S to T. Therefore we inductively define the proposition $f \in S \xrightarrow{\epsilon(d)}_I T$, where always $f : [0,d] \to \mathcal{S}$ satisfying $f(0) = S$ and $f(d) = T$. This definition is in full conformance to the already established clauses of definition of $S \xrightarrow{\epsilon(d)}_I T$. For the sake of brevity, we show only two clauses (the others are more straightforward):

- Whenever $max(0, c-d) \leq c' \leq e' \leq e-d$, we let $f \in [c,e].P \xrightarrow{\epsilon(d)}_I [c',e'].P$ iff $f(x) = [cc(x), ee(x)].P$ for all $x \in [0,d]$ and the functions cc and ee together with $cc(0) = c$, $ee(0) = e$, $cc(d) = c'$ and $ee(d) = e'$ for any $x, y \in [0,d]$ such that $\delta = y - x \geq 0$ satisfy

$$max(0, cc(x) - \delta) \leq cc(y) \leq ee(y) \leq ee(x) - \delta.$$

[27] As can be seen, in the definition of $\approx$ only the endpoints of delay transition are required to be matched.

- Whenever $f \in S \xrightarrow{\epsilon(d)}_I S'$ and $g \in T \xrightarrow{\epsilon(d)}_I T'$, we have $f|g \in S|T \xrightarrow{\epsilon(d)}_I S'|T'$ provided $Sort(f(x)) \cap \overline{Sort(g(x))} = \emptyset$ for all $x \in [0, d[$[28].

We say that $f \in S \overset{\epsilon(d)}{\Longrightarrow}_{Io} S'$ iff $f : [0, d] \to \mathcal{S}$ and $f(d) = S'$, as well as

$$S \overset{\varepsilon}{\Longrightarrow}_\circ S_1 \xrightarrow{\epsilon(d_1)}_I S_1' \overset{\varepsilon}{\Longrightarrow}_\circ \cdots S_k \xrightarrow{\epsilon(d_k)}_I S_k' \overset{\varepsilon}{\Longrightarrow}_\circ S'$$

so that $d_1 + d_2 + \cdots + d_k = d$ and $\forall i \leq k.\ \exists g_i \in S_i \xrightarrow{\epsilon(d_i)}_I S_i'$ such that $\forall \delta \in [0, d_i[.\ f(d_1 + \cdots + d_{i-1} + \delta) = g_i(\delta)$.

$S \overset{\epsilon(d)}{\Longrightarrow}_{Io}$ is defined to be the union of $S \overset{\epsilon(d)}{\Longrightarrow}_{Io} S'$ for all reduced CTR specifications S'.

The refinement $\trianglelefteq_I$ now can be defined by a variant of Definition 3.1 for $\trianglelefteq$, where $\overset{\epsilon(d)}{\Longrightarrow}_{Io}$ is substituted for $\overset{\epsilon(d)}{\Longrightarrow}_\circ$. The principal result of this section (to be proved in the full version) is the following.

Theorem 2 *For any two (reduced) CTR terms S and T $S \trianglelefteq_I T$ if and only if $S \trianglelefteq T$.*

5 (Another) Timed Modal Specifications

In this section we provide yet another semantic variation of (a subset of) CTR, which is based on explicit defining of "necessary" and "possible" modes of transitions, the idea originally appearing in [LT88].

The intuition of the specification refinement in the modal framework is very simple - the more possible and fewer necessary transitions a specification has, the more general it is. This idea has already been adopted to a timed specification formalism in [CGL93, CGL94] resulting in already several times mentioned framework of TMS. Though the TMS model is theoretically very appealing, it turns out that a few of pragmatic properties of the model are a bit unnatural. This applies both to some intuitively desirable refinement examples which are not valid in TMS[29], as well as to the lack of clear operational behaviour intuition due to the having modalities also on the timed transitions.

Here we provide a slight reformulation of the TMS framework (in the light of the developped CTR refinement relation) with the aim of fixing some of its undesirable properties. On the other hand, the theory of A-TMS, sketched in this section, provides yet another semantic variation of the CTR framework, which, as demonstrated by Theorem 3, is consistent with the "standard" development based on $\trianglelefteq$ and explored in Section 3.

Syntactically the A-TMS terms are those of CTR in which every nondeterministic delay prefix is followed by a control action (A-TMS terms have $a_{[c,e]}$,

[28] Here $(f|g)(x)$ is defined to be $f(x)|g(x)$ for all x and $Sort(S) = Sort(0, S)$ for the two-argument $Sort$ function defined in Section 2.

[29] See, for instance, the discussion in Section 3.

$$\frac{--}{nil \xrightarrow{\epsilon(d)} nil} \quad \frac{--}{a.S \xrightarrow{a}_{\Box} S} \quad \frac{S \xrightarrow{a}_m S'}{X \xrightarrow{a}_m S'}[X \stackrel{def}{=} S] \in \mathcal{E} \quad \frac{S \xrightarrow{\epsilon(d)} S'}{X \xrightarrow{\epsilon(d)} S'}[X \stackrel{def}{=} S] \in \mathcal{E}$$

$$\frac{--}{\alpha_{[c,e]}.S \xrightarrow{\epsilon(d)} a_{[c\dot{-}d,e\dot{-}d]}.S} \quad \frac{-- \qquad [d \leq e]}{\tau_{[c,e]}.S \xrightarrow{\epsilon(d)} \tau_{[c\dot{-}d,e-d]}.S} \quad \frac{--}{a_{[0,e]}.S \xrightarrow{a}_{\Diamond} S}$$

$$\frac{S \xrightarrow{a}_m S'}{S+T \xrightarrow{a}_m S'} \quad \frac{T \xrightarrow{a}_m T'}{S+T \xrightarrow{a}_m T'} \quad \frac{S \xrightarrow{\epsilon(d)} S' \;\; T \xrightarrow{\epsilon(d)} T'}{S+T \xrightarrow{\epsilon(d)} S'+T'}$$

$$\frac{S \xrightarrow{a}_m S'}{S|T \xrightarrow{a}_m S'|T} \quad \frac{T \xrightarrow{a}_m T'}{S|T \xrightarrow{a}_m S|T'} \quad \frac{S \xrightarrow{\alpha}_m S' \;\; T \xrightarrow{\overline{\alpha}}_m T'}{S|T \xrightarrow{\tau}_m S'|T'}$$

$$\frac{S \xrightarrow{\epsilon(d)} S' \;\; T \xrightarrow{\epsilon(d)} T'}{S|T \xrightarrow{\epsilon(d)} S'|T'}[Sort(d,S) \cap \overline{Sort(d,T)} = \emptyset]$$

Table 3: Transition rules for A-TMS, $m \in \{\Diamond, \Box\}$

$a_{]c,e]}$ and $\epsilon(d)$ instead of separate $[c,e]$, $]c,e]$ and control action prefixes). We abbreviate $a_{[0,0]}.E$ as $a_{\Box}.E$ (written also as $a.E$) and $a_{[0,\infty]}.E$ as $a_{\Diamond}.E$ [30].

The operational semantics of A-TMS is based on the necessary control transitions $\xrightarrow{a}_{\Box}$, possible control transitions $\xrightarrow{a}_{\Diamond}$ and delays $\xrightarrow{\epsilon(d)}$ [31] [32]. For a fragment of the framework the semantics is summarized in Table 3 (this fragment does not include $\epsilon(d)$ and $a_{]c,e]}$ prefixes, neither restriction and relabelling are considered). The $Sort$ function[33] in the side condition for the parallel composition delay rule is computed, as for CTR terms, following the rules given in Section 2.

In order to define the A-TMS refinement relation let $\mathcal{M}$ be the set of all TMS specifications (over certain sets of specification variables and equations). Let the observational (τ-abstracting) modal transition relations $\stackrel{\mu}{\Longrightarrow}_m$ also be defined in the usual way. The trace set $S \stackrel{\epsilon(d)}{\Longrightarrow}_{\Diamond}$ consists of all traces starting at S and containing interleaving of $\xrightarrow{\epsilon(d_i)}$ and $\xrightarrow{\tau}_{\Diamond}$ transitions, with the total delay amount being d [34]. Also, let μ range over $\Lambda \cup \{\epsilon\}$ (μ is either α, or ϵ).

[30] In the original definition of TMS in [CGL93] the modal prefixes $a_{\Box}.E$ and $a_{\Diamond}.E$ were taken as primitives, and $a_{[c,e]}.E$ was defined as $\epsilon(c).a_{\Diamond}.E + \epsilon(e).a_{\Box}.E$. The $a_{]c,e]}$ prefix here is new, though theoretically it is not principal.

[31] The delays always are in a sense in the possible mode.

[32] A suggested physical interpretation of the necessary and possible transitions is that of transitions of a specification being enabled in a *weak* and *strong* sense. For $a_{[c,e]}.S$ during the first c time units the a transition is not enabled. Afterwards, for the next $c-e$ units this transition is enabled in a weak sense, after what (if it has not been carried out already), it becomes enabled in a strong sense.

[33] This function computes the transition availability in a strong sense.

[34] A formal definition can be given following the style of the trace definition in Section 3. In the same fashion also the relation $\mathcal{R}^t$ for $\mathcal{R} \subseteq \mathcal{M} \times \mathcal{M}$ is defined.

Definition 5.1 *A relation $\mathcal{R} \subseteq \mathcal{M} \times \mathcal{M}$ is a(n) (observational) modal refinement relation, if for any $\langle S, T\rangle \in \mathcal{R}$*

1. $S \stackrel{\mu}{\Longrightarrow}_\Diamond S'$ *implies* $\exists T'.T \stackrel{\mu}{\Longrightarrow}_\Diamond T'$ *so that* $\langle S', T'\rangle \in \mathcal{R}$;
2. $T \stackrel{\mu}{\Longrightarrow}_\Box T'$ *implies* $\exists S'.S \stackrel{\mu}{\Longrightarrow}_\Box S'$ *so that* $\langle S', T'\rangle \in \mathcal{R}$;
3. $f \in S \stackrel{\epsilon(d)}{\Longrightarrow}_\Diamond$ *implies existence of* $g \in T \stackrel{\epsilon(d)}{\Longrightarrow}_\Diamond$ *such that* $\langle f, g\rangle \in \mathcal{R}^t$;
4. $S \stackrel{\tau^\infty}{\longrightarrow}_\Diamond$ *implies* $T \stackrel{\tau^\infty}{\longrightarrow}_\Diamond$.

We say that S is a modal refinement of T, and write $S \lhd_M T$ whenever there exists a refinement relation $\mathcal{R}$ containing $\langle S, T\rangle$ (In other words, $\lhd_M$ is the largest modal refinement relation).

Let $n : \mathcal{M} \to \mathcal{S}$ be a function giving for any A-TMS term S the same term, viewed as a CTR term[35]. The principal theorem of this section says:

Theorem 3 *For any two modal specifications S and T we have $S \lhd_M T$ if and only if $n(S) \unlhd n(T)$.*

Lemma 5.2

1. *Whenever* $S \stackrel{a}{\longrightarrow}_\Diamond S'$, *then* $n(S) \stackrel{a}{\longrightarrow}_\circ n(S')$.
2. *Whenever* $T \stackrel{a}{\longrightarrow}_\Box T'$, *then* $n(T) \stackrel{a}{\longrightarrow} n(T')$.
3. *Whenever* $n(S)(\stackrel{i}{\longrightarrow})^* \stackrel{a}{\longrightarrow} (\stackrel{i}{\longrightarrow})^* Y$, *then* $S \stackrel{a}{\longrightarrow}_\Diamond S'$ *so that* $Y \unlhd n(S')$.
4. *Whenever* $n(T) \stackrel{a}{\longrightarrow} X$, *then* $T \stackrel{a}{\longrightarrow}_\Box T'$ *so that* $n(T') = X$.

To prove $S \lhd_M T \Rightarrow n(S) \unlhd n(T)$ we show that a CTR refinement is the relation $\{\langle X, n(T)\rangle | X \unlhd n(S),\ S \lhd_M T\}$.

The other direction $n(S) \unlhd n(T) \Rightarrow S \lhd_M T$ goes simply by showing that a modal refinement is the relation $\{\langle S, T\rangle | n(S) \unlhd n(T)\}$. In both directions we rely on Lemma 5.2. Details are to be supplied in the full version. $\Box$

6 Decidability

We call a CTR specification *initially integral timed* iff all its initial delay interval prefix bounds are given by integers[36].

Theorem 4 *The refinement relation $\unlhd$ is decidable for initially integral timed CTR specifications.*

The extension of the result of the theorem to allow arbitrary *rational* numbers as the initial delay bounds is straightforward[37]. As a corollary of the theorem we get also the decidability of $\lhd_M$ for TMS.

The main constructions used in the deciding algorithm are those of timed region graph [AD90] and symbolic checking of timed bisimulation [Cer92a, Cer92b], for exposition see also [CGL94, Go94]. The full version will discuss this issue in more detail.

[35] n is an injective mapping of $\mathcal{M}$ into $\mathcal{S}$, introduced for the sake of further convenience.

[36] This requirement is to be applied to the defining equations of all variables which the initial specification relies upon.

[37] Simply by adopting an initial scaling of the time axis.

7 Conclusions

In this paper a mathematically sound and rather general process algebraic framework for refinement of timed system specifications has been proposed, its further fate is to be determined by applicability in CASE studies.

The obtained mathematical results of the coincidence of the three refinement relations $\trianglelefteq$, $\trianglelefteq_I$ and $\triangleleft_M$ show the robustness of the proposed refinement notion - resolving of timed underspecification (nondeterminism). This is a point in favour for trying the proposed calculi on practical examples. It gives an excellent chance for every system designer to understand/imagine the refinement notion in the way he/she likes it most, all choices described here (standard refinement, inline choice, modal specifications) leading to the same formal result.

Still, a somewhat experimental nature of the fourth clauses in all refinement definitions is to be admitted. However, both the theoretical refinement properties and the examples seen so far suggest that this may be a good choice.

A possibility for a further work is to look at incorporating of the introduced timed behaviour underspecification constructs into other process algebraic frameworks (considering disciplines of interprocess communication other than that of CCS may widen the practical applicability of the formalism). It is hoped that the principle of internality of delay nondeterminism, embodied in the semantics of CTR delay interval prefix, should facilitate the possibility of such incorporation.

Though a few comments on the suggested pragmatics of the use of CTR in the real time system design were considered here, there are a few more aspects worth discussing (one of them could be on the pragmatic role of left-open delay interval prefixes).

In a long run one could aim at incorporating both external behaviour and data type refinements into a single framework. For a possible notion of the data type refinement one could look into [BW94] (see also the references therein).

For making CTR a practical specification refinement framework one should work also on practically feasible algorithms for refinement deciding[38]. A possible step into this direction could be a combining of the local validity checking and partition refinement methods for work on deciding "bisimulation - like" properties.

Acknowledgements

The author wants to thank Bengt Jonsson from University of Uppsala and Kim Larsen from University of Aalborg for stimulating discussions and ideas on what a proper notion of operational refinement for timed systems should be.

Thanks are also to BRICS for sponsoring my travel to and attendance at the NWPT'6 conference in Aarhus.

[38] During later stages of the development of the refinement frameworks the possibilities of interactive refinement provers may be considered.

References

[AD90] R. Alur and D. Dill, *Automata for Modelling Real-Time Systems*, in Proceedings of ICALP'90, Lecture Notes in Computer Science No. 443, 1990.

[BW94] R. Back and J. von Wright, *Trace Refinement of Action Systems* in Proceedings of CONCUR'94, Lecture Notes in Computer Science No. 836, 1994.

[CC89] CCITT, *Functional Specification and Description Language (SDL)*, Recommendations Z.100, 1989.

[Cer92a] K. Čerāns, *Decidability of Bisimulation Equivalences for Parallel Timer Processes*, in Proceedings of CAV'92, Lecture Notes in Computer Science No. 663, 1992.

[Cer92b] K. Čerāns, *Algorithmic Problems in Analysis of Real Time System Specifications*, Dr.sc.comp. theses, University of Latvia, 1992.

[CGL93] K. Čerāns, J.Chr. Godskesen and K.G. Larsen, *Timed Modal Specification - Theory and Tools*, in Proceedings of CAV'93, Lecture Notes in Computer Science No. 697, 1993.

[CGL94] K. Čerāns, J.Chr. Godskesen and K.G. Larsen, *Timed Modal Specification - Theory and Tools*, submitted to Formal Aspects of Computing, 1994.

[Che92] L. Chen, *An Interleaving Model for Real Time Systems*, in Proceedings of LFCS'92, Lecture Notes in Computer Science No. 620, 1992.

[Dan92] M. Daniels, *Modelling Real-Time Behaviour with an Interval Time Calculus*, in Proceedings of Conference on Formal Techniques in RT and FT Systems, Nijmegen, 1992

[Dil89] D. Dill, *Trace Theory for Automatic Hierarchical Verification of Speed-Independent Circuits*, ACM Distinguished Dissertations, MIT Press, 1989.

[GMMP89] C. Ghezzi, D. Mandrioli, S. Morasca and M. Pezze *A General Way To Put Time in Petri Nets*, in ACM SIGSOFT Eng. Notes, Vol. 14, No. 3, 1989.

[Go94] J.Chr. Godskesen, *Timed Modal Specifications - A Theory for Verification of Real-Time Concurrent Systems*, Ph.D thesis, Aalborg University, 1994.

[Lam91] L. Lamport, The Temporal Logic of Actions, DEC SRC Technical report No. 79, 1991.

[LT88] K.G. Larsen and B. Thomsen, *A Modal Process Logic*, in Proceedings of the fifth IEEE Symposium on Logic in Computer Science, 1988.

[MF76] P. Merlin and D.J. Farber, *Recoverability of Communication Protocols*, IEEE Trans. on Communication Protocols, Vol. COM-24, No. 9, 1976.

[Mil89] R. Milner, *Communication and Concurrency*, Series in Computer Science, Prentice-Hall International, 1989.

[Wan90] Yi Wang, *Real Time Behaviour of Asynchronous Agents*, in Proceedings of CONCUR'90, Lecture Notes in Computer Science No. 458, 1990.

[Wan91] Yi Wang, *CCS + Time = an Interleaving Model for Real Time Systems*, in Proceedings of ICALP'91, Lecture Notes in Computer Science No. 510, 1991.