# Realizability criteria for compositional MSC[*]

Arjan Mooij, Judi Romijn, and Wieger Wesselink

Technische Universiteit Eindhoven
Department of Mathematics and Computer Science
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
{a.j.mooij,j.m.t.romijn,j.w.wesselink}@tue.nl

**Abstract.** Synthesizing proper implementations for scenario-based specifications is often impossible, due to the distributed nature of implementations. To be able to detect problematic specifications, realizability criteria have been identified, such as non-local choice.
In this work we develop a formal framework to study realizability of compositional MSC [GMP03]. We use it to derive a complete classification of criteria that is closely related to the criteria for MSC from [MGR05].
Comparing specifications and implementations is usually complicated, because different formalisms are used. We treat both of them in terms of a single formalism. Thereto we extend the partial order semantics of [Pra86,KL98] with a way to model deadlocks and with a more sophisticated way to address communication.

## 1 Introduction

For scenario-based specifications of distributed systems (e.g. in terms of Message Sequence Chart, MSC), it is often impossible to synthesize an implementation with exactly the same behavior. This is caused by the distributed nature of implementations. The best-known phenomenon leading to problems is non-local choice [BAL97], but also other criteria [HJ00,Gen05,MGR05] have been proposed to determine realizability of specifications in practice [MG05]. In this work we develop a formal framework to study such criteria for the MSC extension that is called compositional MSC [GMP03,MM01].

Most realizability criteria seem to be tricky formalizations of intuitions about realizability. In contrast, we formally study under what circumstances specifications are trace equivalent to their implementations, and derive a condition that is both necessary and sufficient. From this condition, we derive a complete classification of realizability criteria for compositional MSC. The resulting formal criteria can easily be related to our intuitive criteria in [MGR05].

Several kinds of semantics have been proposed for MSC specifications (e.g. [KL98,Ren99,Hey00,UKM03]), while implementations are typically expressed in terms of finite state machines. To compare specifications and implementations, two different formalisms must then be related, usually via execution traces (in

[*] This research is supported by the NWO under project 016.023.015: "Improving the Quality of Protocol Standards".

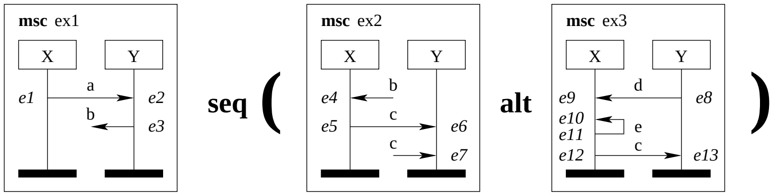


**Fig. 1.** Running example

fact a third formalism). We prefer to use one single formalism for both implementations and specifications, and we want to stay close to the MSC specification formalism. Therefore we use a partial order semantics [Pra86] for our study, and sketch the relation with operational formalisms. In addition to the partial order model in [Pra86,KL98], we introduce a way to model deadlocks and a more sophisticated way to deal with communication.

*Overview* In Section 2 we introduce our partial order model, which we extend with communication in Section 3. These two sections are rather independent from MSC, but they are the basis of the semantics of compositional MSC in Section 4. In Section 5 we define the typical way of synthesizing an implementation; trace equivalence between specifications and such implementations is studied in Section 6. Finally in Section 7 we classify various realizability criteria. The conclusions and further work are presented in Section 8. In the appendix, proofs are listed for the interested reader.

## 2 Extended partial order model

In this section we define a partial order model and extend it with deadlocks, to make it suitable for studying realizability criteria.

### 2.1 Running example

We illustrate our techniques using a running example. Figure 1 contains a (high-level) MSC consisting of the three basic MSCs ex1, ex2 and ex3. It specifies the behavior of process instances $X$ and $Y$, such that first the behavior of ex1 occurs, followed by either the behavior of ex2 or the behavior of ex3. For reference purposes we have included arbitrary event names ($e_1$ to $e_{13}$) in the basic MSCs.

### 2.2 LATERs: LAbeled Transitive Event Relations

As a semantic model of behavior, we introduce the notion of a later, which is an acronym for **la**belled **t**ransitive **e**vent **r**elation. A *later* $(E, <, l)$ is a triple that consists of an event set $E$, a transitive causality relation $<$: $< \subseteq E \times E$ and a labeling function $l : E \to L$ for a given set of labels $L$. The behavior of a later is such that any event $e : e \in E$ models a single action with label $l.e$; the event can occur at most once and it may only occur after all events $f : f < e$ have already occurred. The notion of an event is used to handle multiple occurrences of an action with the same label. Compared to the partial orders in [Pra86], a later is an lposet in which the partial order constraint has been weakened.

In our running example, let laters $p_1$, $p_2$ and $p_3$ correspond to the basic MSCs ex1, ex2 and ex3, such that only the causalities per instance (on each vertical axis) are considered, i.e. without communication. So, $p_1 = (\{e_1, e_2, e_3\}, \{e_2 < e_3\}, l_1)$ and as we will see later on $l_1 = \{e_1 \mapsto !(a, X, Y),\ e_2 \mapsto ?(a, X, Y),\ e_3 \mapsto !(b, Y, X)\}$. The structure of $p_1$ can be visualized as $\boxed{e_1 \qquad e_2 \longrightarrow e_3}$ such that relation $<$ corresponds to the transitive closure of relation $\to$.

In an interleaved execution model where the events are labeled with atomic actions, the maximal behaviors of a partially ordered later are its linearizations. The linearizations of a later $(E, <, l)$ are the execution traces $e_1 \cdot \ldots \cdot e_n$ such that $e_i \in E$ for each index $i$, and for each two indices $i$ and $j$ both $e_i = e_j \Rightarrow i = j$ and $e_i < e_j \Rightarrow i < j$. The linearizations of later $p_1$ are $e_1 \cdot e_2 \cdot e_3$, $e_2 \cdot e_1 \cdot e_3$ and $e_2 \cdot e_3 \cdot e_1$. We prefer to reason about partial orders, because they are better related to MSC and they avoid decomposing each partial order into several over-specific total orders. Another advantage is that true concurrency can be modeled.

The most elementary laters are the empty later, with no events, and the singleton laters, with only one event with a label $k : k \in L$. We introduce the following abbreviations for them:

$$\begin{array}{ll} [\epsilon] = (\emptyset, \emptyset, \emptyset) & \\ {[k]} = (\{e\}, \emptyset, \{[e \mapsto k]\}) & \quad \text{for } k : k \in L \text{ and arbitrary } e \end{array}$$

### 2.3 Isomorphism

The event set of a later is abstract in the sense that a consistent renaming of the events yields a later with the same behavior. This is formalized in the following notion of isomorphism. Laters $(E, <, l)$ and $(E', <', l')$ are *isomorphic*, denoted $(E, <, l) \simeq (E', <', l')$, if there is a bijection $\sim : \sim \subseteq E \times E'$ such that both

- $(\forall e, e' :: e \sim e' \Rightarrow l.e = l'.e')$
- $(\forall e, f, e', f' :: e \sim e' \wedge f \sim f' \Rightarrow (e < f \equiv e' <' f'))$

Relation $\simeq$ is an equivalence relation. In what follows we will hardly mention $\simeq$ explicitly, and implicitly assume that where necessary $\simeq$ has been exploited to obtain suitable laters, e.g. ones that are event disjoint. This conforms to the style used in [KL98].

### 2.4 Elementary later operators

We often need to relate events to the instance (i.e. computational unit or process) in which they occur. We assume a fixed set of *instance names* $I$, and a function[1]

[1] For a later $(E, <, l)$, [HJ00] uses the slightly different function $\phi' : E \to I$, which can be obtained from our later-independent $\phi$ as follows: $\phi'.e = \phi.(l.e)$ .

$\phi : L \rightarrow I$ that maps labels to the instance in which the actions with that label occur. To construct larger laters from the elementary laters, we use the following elementary operators on event disjoint laters (i.e. $E_p \cap E_q = \emptyset$):

$$(E_p, <_p, l_p) \parallel (E_q, <_q, l_q) \;=\; (E_p \cup E_q,\; <_p \cup <_q,\; l_p \cup l_q)$$

$$(E_p, <_p, l_p) \;\circ_S\; (E_q, <_q, l_q) \;=\; (E_p \cup E_q,\; <_p \cup <_{\circ_S} \cup <_q,\; l_p \cup l_q)$$
$$\text{where } <_{\circ_S} = E_p \times E_q$$

$$(E_p, <_p, l_p) \;\circ_W\; (E_q, <_q, l_q) \;=\; (E_p \cup E_q,\; (<_p \cup <_{\circ_W} \cup <_q)^+,\; l_p \cup l_q)$$
$$\text{where } <_{\circ_W} = \{(e, f) \mid e, f :\; e \in E_p \;\wedge\; f \in E_q \;\wedge\; \phi.(l_p.e) = \phi.(l_q.f)\}$$

Operator $\parallel$ denotes parallel composition, and operators $\circ_S$ and $\circ_W$ denote strong and weak sequential composition, respectively. These operators are associative and they have unit element $[\epsilon]$. Since parallel composition is also commutative, we can use $\parallel$ as a quantifier.

In our running example, $\phi.(!(a, X, Y)) = X$ and $\phi.(?(a, X, Y)) = Y$. Let laters $p_4$ and $p_5$ be defined as $p_4 = p_1 \circ_W p_2$ and $p_5 = p_1 \circ_W p_3$. The structure of $p_5$ is visualized as $\boxed{e_1 \rightarrow e_9 \rightarrow e_{10} \rightarrow e_{11} \rightarrow e_{12} \quad e_2 \rightarrow e_3 \rightarrow e_8 \rightarrow e_{13}}$ .

## 2.5 Deadlocks

A later $(E, <, l)$ contains a *deadlock* if there is an event $e :\; e \in E$ such that $e < e$. Conversely, a later is *deadlock-free* if the (transitive) causality relation is a strict partial order, i.e. the conjunction of the following holds:

- irreflexive: $(\forall e :: \neg(e < e))$
- asymmetric: $(\forall e, f :: \neg(e < f \;\wedge\; f < e))$
- transitive: $(\forall e, f, g :: e < f \;\wedge\; f < g \;\Rightarrow\; e < g)$

The definitions of deadlock and deadlock-free are consistent, since asymmetry implies irreflexivity, and transitivity plus irreflexivity implies asymmetry. In particular, all laters that can be obtained from the elementary laters using the elementary later operators are deadlock-free.

For example, consider later $p'_5$ (to be defined in Section 3) with the following structure: $\boxed{e_1 \rightarrow e_2 \rightarrow e_3 \rightarrow e_8 \rightarrow e_9 \rightarrow e_{10} \leftrightarrows e_{11} \rightarrow e_{12} \rightarrow e_{13}}$ . In this later there is a circular dependency between events $e_{10}$ and $e_{11}$. From the transitivity of relation $<$ it follows that $e_{10} < e_{10}$, hence $e_{10}$ is a deadlock.

The interpretation of the causality relation is such that the set of events "behind any deadlock" cannot occur either. We define the set of deadlocked events $\Delta$ for a later $(E, <, l)$ as follows:

$$\Delta.(E, <, l) = \{f \mid e, f :\; e \in E \;\wedge\; f \in E \;\wedge\; e < e \;\wedge\; e < f\}$$

In our example we obtain $\Delta.p'_5 = \{e_{10}, e_{11}, e_{12}, e_{13}\}$, and hence events $e_1$, $e_2$, $e_3$, $e_8$ and $e_9$ are the only events that can occur in later $p'_5$.

### 2.6 Prefix

A natural way to compare laters is to compare their possible behaviors. If all possible behaviors of a later $p$ are contained[2] in the possible behaviors of a later $q$, we call $p$ a prefix of $q$. To determine whether $p$ is a prefix of $q$, we only need to consider the deadlock-free part of $p$. If $p$ is a prefix of $q$, then (1) $p$ may contain fewer events than $q$, (2) on this smaller event set, $p$ may contain more causalities than $q$, (3) $q$'s labeling of events is respected by $p$, and (4) for each event that is in both $p$ and $q$, all events that precede the event in $q$ are also in $p$.

Formally, later $p$ is a *prefix* of later $q$, denoted $p \preceq q$, if for some laters $(E_p, <_p, l_p) \simeq p$ and $(E_q, <_q, l_q) \simeq q$ the following four conditions hold:

1. $\overline{E_p} \subseteq E_q$
2. $<_q \cap (\overline{E_p} \times \overline{E_p}) \subseteq <_p$
3. $l_p \cap (\overline{E_p} \times L) = l_q \cap (\overline{E_p} \times L)$
4. $(\forall e, f :: e <_q f \wedge f \in \overline{E_p} \Rightarrow e \in \overline{E_p})$

where $\overline{E_p} = E_p \backslash \mathit{\Delta}.(E_p, <_p, l_p)$

In the running example several prefix relations hold, such as $p_1 \preceq p_4$ and $p_1 \preceq p_5$.

As a corollary of $p \preceq q$, we have $\overline{E_p} \subseteq \overline{E_q}$ for $\overline{E_q} = E_q \backslash \mathit{\Delta}.(E_q, <_q, l_q)$. Prefix order $\preceq$ is a pre-order (i.e. reflexive and transitive) with smallest element $[\epsilon]$. Some typical prefixes are $p \preceq p \| q$, $q \preceq p \| q$, $p \preceq p \circ_S q$ and $p \preceq p \circ_W q$. In comparison with [KL98], our definition is more explicit, it can deal with deadlocks and it allows $<_q \cap (\overline{E_p} \times \overline{E_p})$ to be strictly smaller than $<_p$.

Parallel composition is monotonic in both arguments, while both kinds of sequential composition are only monotonic in their second argument (since deadlocks are invisible). In general, sequential composition is not monotonic in its first argument. For example, let $p = [\epsilon]$, $q = (\{e\}, \{e < e\}, \{e \mapsto k\})$ and $r = [k']$ such that $\phi.k = \phi.k'$. Using $\phi.k = \phi.k'$, both kinds of sequential composition yield $p \circ r = r$ and $q \circ r \doteq q$. Although $p \preceq q$, we do not have $p \circ r \preceq q \circ r$, because $r \not\preceq q$. This observation has directed our study in Section 6.2 towards an action-prefix alike operator instead of a full sequential composition operator.

A special kind of prefix is a causality extension:

$$< \subseteq <' \Rightarrow (E, <', l) \preceq (E, <, l)$$

As an example consider later $p_5'$, which is a causality extension of later $p_5$.

### 2.7 Projection

To restrict the set of events of a later, we define a projection operator $\pi$ that restricts a later to the events in instance $i$ as follows:

$$\begin{gathered}\pi_i.(E, <, l) = (F, < \cap (F \times F), l \cap (F \times L)) \\ \text{where } F = \{e \mid e : e \in E \wedge \phi.(l.e) = i\}\end{gathered}$$

Its relation with parallel composition is $p \preceq (\| i : i \in I : \pi_i.p)$, and it is monotonic with respect to causality extensions:

$$< \subseteq <' \Rightarrow \pi_i.(E, <', l) \preceq \pi_i.(E, <, l)$$

[2] In an interleaved execution model this corresponds to trace inclusion.

### 2.8 Sets of laters

Usually a single later cannot describe all possible behavior of a system. Thereto we study a set of laters (which is the notion of process in [Pra86], and pomset in [KL98]), which represents the set of behaviors of the individual laters. We lift each elementary later operator $\oplus$ and the projection operator $\pi$ as follows:

$$\begin{aligned} P \oplus Q &= \{p \oplus q \mid p, q : \ p \in P \ \wedge \ q \in Q\} \\ \pi_i.P &= \{\pi_i.p \mid p : \ p \in P\} \end{aligned}$$

To lift the prefix order $\preceq$, we define order $\sqsubseteq$ as follows:

$$P \sqsubseteq Q \ \equiv \ (\forall p : p \in P : \ (\exists q : q \in Q : \ p \preceq q))$$

Order $\sqsubseteq$ is a pre-order with smallest element $\emptyset$. Like before, parallel composition is monotonic in both arguments, while both kinds of sequential composition are only monotonic in their second argument. Relation $\doteq$ defined as

$$P \doteq Q \ \equiv \ P \sqsubseteq Q \ \wedge \ Q \sqsubseteq P$$

is an equivalence relation. Equivalence $P \doteq Q$ denotes that $P$ and $Q$ have the same sets of deadlock-free prefixes, which means that they are trace equivalent.

## 3 Asynchronous communication

In this section we develop an operator that introduces in a later the causalities that correspond to asynchronous message communication. To model distributed systems with communication via message passing, some labels are used to denote sending or receiving a message. The most liberal causalities are obtained by matching sends and receipts in their order of occurrence. This does not require that messages with identical names are communicated in FIFO order.

### 3.1 Label-wise trichotomy

To match events properly, we need to determine the order in which events with identical labels occur. For simplicity reasons, we assume for each label that the events with that label are totally ordered; at least, in the deadlock-free part of the later. Since this deadlock-free part is strict partially ordered, we only need trichotomy (or comparability) for events with identical labels. For notational convenience, we require this property for the whole later and for all labels.

The *label-wise trichotomy* property $T$ is defined as follows:

$$\begin{aligned} &T.P \ \equiv \ (\forall p : p \in P : \ T.p) \\ &T.(E, <, l) \ \equiv \ (\forall e, f :: \ l.e = l.f \ \Rightarrow \ e = f \ \vee \ e < f \ \vee \ f < e) \end{aligned}$$

As we will see in Section 4, this only imposes a few, acceptable restrictions to MSCs. This property is maintained under causality extensions and event restrictions, it holds for the elementary laters, and it is maintained under sequential composition; only for a parallel composition $(E_p, <_p, l_p) \parallel (E_q, <_q, l_q)$ label-disjointness is required, i.e. $(\forall e, f : e \in E_p \ \wedge \ f \in E_q : \ l_p.e \neq l_q.f)$.

### 3.2 Communication causalities

We define operator $\Gamma.p$, which introduces the communication causalities in a later $p$. For compositional MSC, we must also address communication between two sequentially composed laters. Thereto we introduce an extra parameter $t$ to denote the entire preceding behavior of later $p$ in terms of a later.

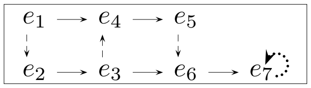
For each message $m$, we must ensure that each receipt event (with label $?m$) is preceded by the corresponding/matching send event (with label $!m$). In case there are more receive events than send events, these remaining receipt events are turned into deadlocks. Thus we obtain (provided $T.t$ and $T.P$ hold):

$$\begin{aligned}
&\Gamma^t.P = \{\Gamma^t.p \mid p:\ p \in P\}\\
&\Gamma^t.(E, <_b, l) = (E,\ (<_b \cup <_c)^{+} \cup <_d,\ l)\\
&\quad \text{where } <_c = <'_c\ \cap (E \times E) \text{ and } <_d = <'_d\ \cap (E \times E)\\
&\quad \text{and } (E', <', l') = t \circ_W (E, <_b, l) \text{ and } \overline{E'} = E' \backslash \Delta.(E', <', l')\\
&\quad \text{and } <'_c = \{(e, f) \mid e, f, m:\ e \in \overline{E'}\ \wedge\ f \in \overline{E'}\ \wedge\ l'.e = !m\ \wedge\ l'.f = ?m\ \wedge\\
&\qquad\qquad\qquad (\#g :: g <' e\ \wedge\ l'.g = !m)\ =\ (\#g :: g <' f\ \wedge\ l'.g = ?m)\}\\
&\quad \text{and } <'_d = \{(f, f) \mid f, m:\ f \in \overline{E'}\ \wedge\ l'.f = ?m\ \wedge\\
&\qquad\qquad\qquad (\#g :: g \in \overline{E'}\ \wedge\ l'.g = !m)\ \leq\ (\#g :: g <' f\ \wedge\ l'.g = ?m)\}
\end{aligned}$$

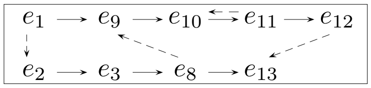
In this definition, first an auxiliary later $(E', <', l')$ is computed as the sequential composition of $t$ and $(E, <_b, l)$. Then causalities $<'_c$ are defined for the matching communications, and causalities $<'_d$ are defined for the deadlocked receipt events. Finally, only the causalities on events $E$ (i.e. not on events from previous behavior $t$) are added to later $(E, <_b, l)$.

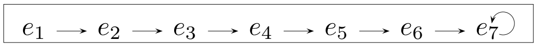
For the running example, we define later $p'_4 = \Gamma^{\emptyset}.p_4$ and $p'_5 = \Gamma^{\emptyset}.p_5$. When visualizing $p'_4$ and $p'_5$, we add the additional communication causalities according to $<'_c$ with dashed arrows, and the additional deadlock causality for unmatched receipts ($<'_d$) with a dotted arrow as follows:

$p'_4$:



$p'_5$:



For $p'_4$ this then boils down to: .

For $p'_5$, the result was already visualized in Section 2.

The role of parameter $t$ of $\Gamma$ is illustrated in the following important property of sequential composition (see also Section 6):

$$\Gamma^t.(\{p\} \circ_W Q)\ \ \doteq\ \ \Gamma^t.(\{p\}\ \circ_W\ \Gamma^{t \circ_W p}.Q)$$

Since $\Gamma$ is a causality extension, it maintains predicate $T$. However, $\Gamma$ can introduce deadlocks. The following are some other properties of $\Gamma$:

$$\begin{array}{ll}
(\textit{shrinking}) & \Gamma^t.p\ \preceq\ p\\
(\textit{idempotence}) & \Gamma^t.p\ =\ \Gamma^t.(\Gamma^t.p)\\
(\textit{monotonicity}) & p \preceq q\ \ \Rightarrow\ \ \Gamma^t.p\ \preceq\ \Gamma^t.q
\end{array}$$

These properties can even be generalized to sets of laters.

## 4 Semantics of compositional MSC

Using the preceding concepts, we define a semantics of compositional MSC as an extension of the MSC semantics of [KL98]. For simplicity reasons, we delay the introduction of the communication causalities; in Section 6 we will show how they can be introduced earlier (like in [KL98]). We start by giving the semantics of basic MSC, then the semantics of high-level MSC, and finally we complete this semantics by including the communication causalities.

### 4.1 Basic MSC

The semantics (without communication) of basic MSC $B$ in instance-oriented textual representation [Ren99] is defined as a later $M_{bmsc}[\![B]\!]$ as follows:

$$\begin{aligned} M_{bmsc}[\![\langle\,\rangle]\!] &= [\epsilon] \\ M_{bmsc}[\![\textbf{inst}\; i; S\;\textbf{endinst}; B]\!] &= M_{inst}[\![S]\!](i) \parallel M_{bmsc}[\![B]\!] \end{aligned}$$

$$\begin{aligned} M_{inst}[\![\langle\,\rangle]\!](i) &= [\epsilon] \\ M_{inst}[\![a; S]\!](i) &= M_{inst}[\![a]\!](i) \;\circ_S\; M_{inst}[\![S]\!](i) \\ M_{inst}[\![\textbf{in}\; n\;\textbf{from}\; j]\!](i) &= [?(n,j,i)] \\ M_{inst}[\![\textbf{out}\; n\;\textbf{to}\; j]\!](i) &= [!(n,i,j)] \\ M_{inst}[\![\textbf{local}\; b]\!](i) &= [b(i)] \\ M_{inst}[\![\textbf{co}\;\langle\,\rangle\;\textbf{endco}]\!](i) &= [\epsilon] \\ M_{inst}[\![\textbf{co}\; a; C\;\textbf{endco}]\!](i) &= M_{inst}[\![a]\!](i) \parallel M_{inst}[\![\textbf{co}\; C\;\textbf{endco}]\!](i) \end{aligned}$$

Function $\phi$ can then be defined as follows: $\phi.(?(n,j,i)) = i$, $\phi.(!(n,i,j)) = i$ and $\phi.(b(i)) = i$ . By construction, each later $M_{bmsc}[\![...]\!]$ is a strict partial order.

To ensure that predicate $T$ is satisfied, we assume that no instance name occurs more than once per bMSC [Ren99], and we require that in each co-region the events are label disjoint. The interest in co-regions is usually very limited (they are completely excluded in [HJ00,GMP03]), so this is no severe restriction. The unrealistic assumption that for each message name there is at most one send event and at most one receipt event per bMSC [KL98], is not required here.

### 4.2 High-level MSC

The semantics (without communication) of high-level MSC $A$ in textual representation is defined as a set of laters $M_{hmsc}[\![A]\!]$ as follows:

$$\begin{aligned} M_{hmsc}[\![\textbf{empty}]\!] &= \{[\epsilon]\} \\ M_{hmsc}[\![\textbf{msc}\; name; B\;\textbf{endmsc}]\!] &= \{M_{bmsc}[\![B]\!]\} \\ M_{hmsc}[\![A\;\textbf{seq}\; B]\!] &= M_{hmsc}[\![A]\!] \;\circ_W\; M_{hmsc}[\![B]\!] \\ M_{hmsc}[\![A\;\textbf{alt}\; B]\!] &= M_{hmsc}[\![A]\!] \;\cup\; M_{hmsc}[\![B]\!] \end{aligned}$$

By construction, each later in $M_{hmsc}[\![...]\!]$ is a strict partial order, and satisfies predicate $T$. We do not explicitly address iteration, since it is just repeated sequential composition.

### 4.3 MSC

Finally we introduce the causalities imposed by communication:

$$\begin{aligned} M_{msc}[\![A]\!] &= M^{\emptyset}_{msc}[\![A]\!] \\ M^{t}_{msc}[\![A]\!] &= \Gamma^{t}.M_{hmsc}[\![A]\!] \end{aligned}$$

This is a proper definition since $M_{hmsc}[\![A]\!]$ satisfies predicate $T$. By construction, predicate $T$ also holds for $M^{t}_{msc}[\![A]\!]$. Note that the application of $\Gamma^{t}$ may introduce deadlocks, which violate the strict partial order property. This illustrates one of the reasons for our extended partial order semantics.

Using the example laters from Sections 2 and 3, the semantics of the MSC in Figure 1 corresponds to $\Gamma^{\emptyset}.(\{p_1\} \circ_W (\{p_2\} \cup \{p_3\}))$, which simplifies via $\{\Gamma^{\emptyset}.(p_1 \circ_W p_2), \Gamma^{\emptyset}.(p_1 \circ_W p_3)\}$ into $\{p'_4, p'_5\}$. These two laters represent the possibility of either performing ex1 followed by ex2, or ex1 followed by ex3.

In [GMP03] there is a restriction that receive events in bMSCs may not be matched to send events in future bMSCs. In [MM01] an extension is proposed that drops this restriction. We consider the extension, since the original restriction conflicts with elegant rules, like sequential composition of two bMSCs being equal to simply connecting the instance axis [Ren99].

## 5 Implementations

In this section we explain how specifications are implemented. The difference between a specification and an implementation is that a specification describes behavior in terms of all instances, while an implementation describes behavior in terms of each individual instance. Thus an implementation for an instance can be represented by a set of laters that contain events of that instance only.

To synthesize an implementation, the specification is decomposed according to the instances. The joint execution behavior of an implementation is obtained by recomposing the instances. We do not consider the unusual implementation with message parameters proposed in [Gen05], which effectively boils down to renaming the messages and shifting the moments of choice. In such an implementation, additional parameters in a request message are sometimes used to fix the choice that should made by the receiver of the request.

### 5.1 Decomposition

The typical decomposition $D$ of a set of laters $M$ to its instances is:

$$D.M \;=\; \{[i \mapsto \pi_i.M] \mid i : \; i \in I\}$$

In this set, each instance name is mapped to the corresponding projection of $M$. Since projection is an event restriction, predicate $T$ is maintained.

For our running example, the decomposition of the laters, $D.\{p'_4, p'_5\}$, yields the following: $\{\, [X \mapsto \{\, \boxed{e_1 \longrightarrow e_4 \longrightarrow e_5} \,,\, \boxed{e_1 \longrightarrow e_9 \longrightarrow e_{10} \rightleftarrows e_{11} \longrightarrow e_{12}} \,\}]$, $[Y \mapsto \{\, \boxed{e_2 \longrightarrow e_3 \longrightarrow e_6 \longrightarrow e_7 \circlearrowleft} \,,\, \boxed{e_2 \longrightarrow e_3 \longrightarrow e_8 \longrightarrow e_{13}} \,\}]\,\}$.

Let us briefly investigate what might be lost by decomposition. For a singleton set $\{(E,<,l)\}$, note that $E$ and $l$ are partitioned per instance, and hence only the causalities between different instances are lost. For each later in a larger set $M$, also the link between its projections in the different instances is lost.

### 5.2 Recomposition

To study the joint execution behavior of the decompositions, the decomposition has to be recomposed. Using the definition from the previous section, the typical recomposition $R$ of a decomposition becomes:

$$R^t.\{[i \mapsto \pi_i.M] \mid i:\ i \in I\} \ = \ \Gamma^t.(\|i : i \in I :\ \pi_i.M)$$

This is a proper definition provided $T.M$ holds, since $T$ is maintained under parallel composition with disjoint labels. The projections are label-disjoint, since for each label $k$ all events with that label belong to one instance, viz. $\phi.k$ .

We emphasize that $R^t \circ D$, where $\circ$ denotes function composition, is not monotonic with respect to $\sqsubseteq$. For causality extensions like $\Gamma^t$, we have:

$$(R^t \circ D).(\Gamma^t.P) \ \ \sqsubseteq \ \ (R^t \circ D).P$$

### 5.3 Implementations in operational formalisms

Using our later representation, implementations in operational formalisms can easily be obtained. In an interleaved execution model where the labels denote atomic actions, the maximal behaviors of a single later are the linearizations of the maximal deadlock-free prefix. The set of maximal behaviors of a set of laters is the union of the linearizations of the individual laters. In turn, linearizations can easily be transformed to process algebraic expressions using the delayed choice operator [BM95]. The implementation of our running example corresponds to the following CSP-style implementation:

$$\begin{array}{llllll} X: & !a & \cdot\,(?b \cdot !c & + & ?d & ) \\ Y: & ?a \cdot !b & \cdot\,(?c & + & !d \cdot ?c & ) \end{array}$$

## 6 Relation between specification and implementation

In this section, we investigate whether compositional MSC specifications are trace equivalent to their implementations, i.e. for all $A$ and $t$:

$$M^t_{msc}[\![A]\!] \ \ \doteq \ \ (R^t \circ D).M^t_{msc}[\![A]\!]$$

### 6.1 The implementation contains the specification

In this section we show that the specification is contained in the implementation, i.e. for all $A$ and $t$: $M^{t}_{msc}[\![A]\!] \sqsubseteq (R^{t} \circ D).M^{t}_{msc}[\![A]\!]$. It can be proved as follows:

$$\begin{array}{ll}
 & (R^{t} \circ D).M^{t}_{msc}[\![A]\!] \\
= & \quad \{\text{definition of } R^{t} \circ D\} \\
 & \mathit{\Gamma}^{t}.(\|i : i \in I :\ \pi_{i}.M^{t}_{msc}[\![A]\!]) \\
\sqsupseteq & \quad \{\text{property of } \pi \text{ and } \|\text{; monotonicity of } \mathit{\Gamma}\} \\
 & \mathit{\Gamma}^{t}.M^{t}_{msc}[\![A]\!] \\
= & \quad \{\text{definition of } M^{t}_{msc}[\![A]\!]\text{; idempotence of } \mathit{\Gamma}\} \\
 & M^{t}_{msc}[\![A]\!]
\end{array}$$

### 6.2 The specification contains the implementation

In this section we derive conditions under which the implementation is contained in the specification, i.e. for all $A$ and $t$: $(R^{t} \circ D).M^{t}_{msc}[\![A]\!] \sqsubseteq M^{t}_{msc}[\![A]\!]$. We will set up an inductive argument based on the structure of the high-level MSC. Thereto we assume that the following rewrite rules have been applied:

$$\begin{array}{lcl}
(\textbf{empty})\ \textbf{seq}\ C & \rightarrow & C \\
(A\,\textbf{seq}\,B)\ \textbf{seq}\ C & \rightarrow & A\ \textbf{seq}\ (B\,\textbf{seq}\,C) \\
(A\,\textbf{alt}\,B)\ \textbf{seq}\ C & \rightarrow & (A\,\textbf{seq}\,C)\ \textbf{alt}\ (B\,\textbf{seq}\,C)
\end{array}$$

These rules do not change the occurrences of choice, but they ensure that the first argument of sequential composition is just a single bMSC. Using the property of $\mathit{\Gamma}$ and $\circ_{W}$ in Section 3, we derive an alternative characterization of $M^{t}_{msc}[\![...]\!]$ in which communication is addressed earlier (like in [KL98]):

$$\begin{array}{ll}
M^{t}_{msc}[\![\textbf{msc}\ \mathit{name}; A\ \textbf{endmsc}]\!] & = M^{t}_{msc}[\![\textbf{msc}\ \mathit{name}; A\ \textbf{endmsc seq empty}]\!] \\
M^{t}_{msc}[\![\textbf{empty}]\!] & = \{[\epsilon]\} \\
M^{t}_{msc}[\![\textbf{msc}\ \mathit{name}; A\ \textbf{endmsc seq}\ B]\!] & \doteq \mathit{\Gamma}^{t}.(\{M_{bmsc}[\![A]\!]\} \circ_{W} M^{t \circ_{W} M_{bmsc}[\![A]\!]}_{msc}[\![B]\!]) \\
M^{t}_{msc}[\![A\ \textbf{alt}\ B]\!] & = M^{t}_{msc}[\![A]\!] \cup M^{t}_{msc}[\![B]\!]
\end{array}$$

**Empty** This is the base case, which has a very simple proof:

$$\begin{array}{ll}
 & (R^{t} \circ D).M^{t}_{msc}[\![\textbf{empty}]\!] \\
= & \quad \{\text{alternative characterization}\} \\
 & (R^{t} \circ D).\{[\epsilon]\} \\
= & \quad \{\text{calculus}\} \\
 & \{[\epsilon]\} \\
= & \quad \{\text{alternative characterization}\} \\
 & M^{t}_{msc}[\![\textbf{empty}]\!]
\end{array}$$

**Sequential composition** This inductive case can be proved as follows:

$$\begin{array}{ll}
 & (R^t \circ D).M^t_{msc}\llbracket \mathbf{msc}\, name; A\, \mathbf{endmsc}\ \ \mathbf{seq}\, B\rrbracket \\
\doteq & \{\text{alternative characterization}\} \\
 & (R^t \circ D).(\Gamma^t.(\{M_{bmsc}\llbracket A\rrbracket\} \circ_W M^{t\circ_W M_{bmsc}\llbracket A\rrbracket}_{msc}\llbracket B\rrbracket)) \\
\sqsubseteq & \{\text{monotonicity}\} \\
 & (R^t \circ D).(\{M_{bmsc}\llbracket A\rrbracket\} \circ_W M^{t\circ_W M_{bmsc}\llbracket A\rrbracket}_{msc}\llbracket B\rrbracket) \\
\doteq & \{\bullet \text{ see below}\} \\
 & \Gamma^t.(\{M_{bmsc}\llbracket A\rrbracket\} \circ_W (R^{t\circ_W M_{bmsc}\llbracket A\rrbracket} \circ D).M^{t\circ_W M_{bmsc}\llbracket A\rrbracket}_{msc}\llbracket B\rrbracket) \\
\doteq & \{\text{induction hypothesis, monotonicity of } \Gamma \text{ and } \circ_W\} \\
 & \Gamma^t.(\{M_{bmsc}\llbracket A\rrbracket\} \circ_W M^{t\circ_W M_{bmsc}\llbracket A\rrbracket}_{msc}\llbracket B\rrbracket) \\
\doteq & \{\text{alternative characterization}\} \\
 & M^t_{msc}\llbracket \mathbf{msc}\, name; A\, \mathbf{endmsc}\ \ \mathbf{seq}\, B\rrbracket
\end{array}$$

The step marked • follows from the following rule, where $m$ denotes a later that does not order events in different instances, and $M$ denotes a set of laters:

$$(R^t \circ D).(\{m\} \circ_W M) \ \doteq\ \Gamma^t.(\{m\} \circ_W (R^{t\circ_W m} \circ D).M)$$

**Alternative composition** This inductive case can be proved as follows:

$$\begin{array}{ll}
 & (R^t \circ D).M^t_{msc}\llbracket A\, \mathbf{alt}\, B\rrbracket \\
= & \{\text{alternative characterization}\} \\
 & (R^t \circ D).(M^t_{msc}\llbracket A\rrbracket \cup M^t_{msc}\llbracket B\rrbracket) \\
\sqsubseteq & \{\blacktriangle \text{ see below}\} \\
 & (R^t \circ D).M^t_{msc}\llbracket A\rrbracket \ \cup\ (R^t \circ D).M^t_{msc}\llbracket B\rrbracket \\
\doteq & \{\text{induction hypothesis (twice)}\} \\
 & M^t_{msc}\llbracket A\rrbracket \ \cup\ M^t_{msc}\llbracket B\rrbracket \\
= & \{\text{alternative characterization}\} \\
 & M^t_{msc}\llbracket A\, \mathbf{alt}\, B\rrbracket
\end{array}$$

The step marked ▲ is not only a sufficient condition, but also a necessary one. Since it does not hold for each MSC, we will study it further.

### 6.3 Safe choice

In Section 7 we will relate various realizability criteria to condition ▲ before. In this section, we first strengthen this condition into a more convenient one. By definition of $R^t \circ D$, it is equivalent to:

$$\Gamma^t.(\|i :: \pi_i.(M^t_{msc}\llbracket A\rrbracket \cup M^t_{msc}\llbracket B\rrbracket)) \preceq \Gamma^t.(\|i :: \pi_i.M^t_{msc}\llbracket A\rrbracket) \cup \Gamma^t.(\|i :: \pi_i.M^t_{msc}\llbracket B\rrbracket)$$

Or formulated differently, for each function $f :: [I \to (M^t_{msc}\llbracket A\rrbracket \cup M^t_{msc}\llbracket B\rrbracket)]$ representing the chosen later per instance, (at least) one of the following holds (where $g$ and $h$ denote functions):

$$(\exists g : g :: [I \to M^t_{msc}\llbracket A\rrbracket] :\ \Gamma^t.(\|i ::\ \pi_i.f_i)\ \ \preceq\ \ \Gamma^t.(\|i ::\ \pi_i.g_i))$$

$$(\exists h : h :: [I \to M^t_{msc}\llbracket B\rrbracket] :\ \Gamma^t.(\|i ::\ \pi_i.f_i)\ \ \preceq\ \ \Gamma^t.(\|i ::\ \pi_i.h_i))$$

Checking this condition is quite involved in practice, since arbitrary combinations of projected laters (i.e. from both $M^t_{msc}\llbracket A\rrbracket$ and $M^t_{msc}\llbracket B\rrbracket$) need to be considered. To reduce the number of combinations, we strengthen[3] this condition for non-empty set $I$ into what we call the *safe choice* property: there exists an instance $k$ such that for each instance $j : j \neq k$ both

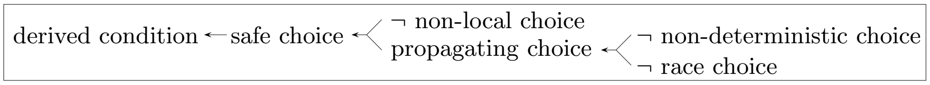
- $\forall\ g :: [I \rightarrow M^t_{msc}\llbracket A \rrbracket],\ n :\ n \in \pi_j.M^t_{msc}\llbracket B \rrbracket\ \wedge\ \{n\} \not\sqsubseteq \pi_j.M^t_{msc}\llbracket A \rrbracket$:
$$\Gamma^t.((\|i : i \neq j :\ \pi_i.g_i)\ \|\ n)\ \preceq\ \Gamma^t.(\|i : i \neq j :\ \pi_i.g_i)$$
- $\forall\ h :: [I \rightarrow M^t_{msc}\llbracket B \rrbracket],\ m :\ m \in \pi_j.M^t_{msc}\llbracket A \rrbracket\ \wedge\ \{m\} \not\sqsubseteq \pi_j.M^t_{msc}\llbracket B \rrbracket$:
$$\Gamma^t.((\|i : i \neq j :\ \pi_i.h_i)\ \|\ m)\ \preceq\ \Gamma^t.(\|i : i \neq j :\ \pi_i.h_i)$$

Later $n : n \in \pi_j.M^t_{msc}\llbracket B \rrbracket \wedge \{n\} \not\sqsubseteq \pi_j.M^t_{msc}\llbracket A \rrbracket$ of instance $j$ denotes a later from MSC $B$ that is no prefix of any behavior on the other side of the choice, i.e. from any later from MSC $A$. Note that behaviors occuring both in MSC $A$ and MSC $B$ are no problem for the choice between $A$ and $B$.

The advantage of this condition is that in the left-hand side of the $\preceq$, the combinations of projected laters contain only one later $n$ from $B$, while all other laters are from $A$. Furthermore, it is less symmetric due to instance $k$ and condition $j \neq k$, see non-local choice below. Finally, we stress that this condition is stronger than the previous one, see non-deterministic choice below.

## 7 Realizability criteria

The safe choice property of the previous section implies that the specification and the implementation are trace equivalent; otherwise the specification may not be realizable. In this section we convert the realizability criteria from [MGR05] to high-level MSCs with binary choice, and generalize them to compositional MSC with co-regions. We first depict how the criteria are classified in comparison with safe choice and the original derived condition from the previous section:



### 7.1 Non-local choice

A choice between two MSCs is local if at most one instance has initiative in these MSCs; otherwise several instances can independently start executing different MSCs. An instance has initiative in an MSC if some first event of the instance is labeled with either an internal action, or sending a message, or receiving a message that was sent before the choice. The choice in our running example is non-local, since due to events $e_4$ and $e_8$ both $X$ and $Y$ have initiative.

Non-local choice follows naturally from safe choice, and in particular from its $\preceq$-terms. Observe that a later $n$ is likely to be problematic if for each label-disjoint later $x$ we have $\Gamma^t.(x\|n) \npreceq \Gamma^t.x$. This condition follows from $\Gamma^t.n \npreceq [\epsilon]$,

[3] The proof of this strengthening step is quite involved.

which means that later $n$ contains an initiating event. Due to condition $j \neq k$ in the definition of safe choice, only instance $k$ may have initiative, i.e. no two different instances, say $i$ and $j$, may have initiative. This leads to the *non-local choice* criterion:

$$\begin{array}{l}(\exists i,j,m,n :: \ i \neq j \ \wedge m \in \pi_i.M^t_{\mathit{msc}}[\![A]\!] \ \wedge \ \{m\} \not\sqsubseteq \pi_i.M^t_{\mathit{msc}}[\![B]\!] \ \wedge \ \varGamma^t.m \not\preceq [\epsilon] \\ \qquad\qquad\qquad\qquad\quad \wedge\, n \in \pi_j.M^t_{\mathit{msc}}[\![B]\!] \ \wedge \ \{n\} \not\sqsubseteq \pi_j.M^t_{\mathit{msc}}[\![A]\!] \ \wedge \ \varGamma^t.n \not\preceq [\epsilon] \quad )\end{array}$$

The difference with other variants of non-local choice in [BAL97,HJ00,MGR05] is in our first two conjuncts on both $m$ and $n$, where we ensure that safe choice is violated.

### 7.2 Propagating choice

Absence of non-local choice is not sufficient to guarantee safe choice. It does guarantee that there is at most one instance that determines the choice, viz. instance $k$ in the definition of safe choice. The other instances $j$ have no initiative and hence their chosen laters $n$ are characterized by $\varGamma^t.n \preceq [\epsilon]$. What remains to guarantee safe choice is that the other instances can resolve the choice, which is characterized by the *propagating choice* property (see also [MGR05]): for each instance $j$ both

– $\forall \ g :: [I \rightarrow M^t_{\mathit{msc}}[\![A]\!]],\ n :\ n \in \pi_j.M^t_{\mathit{msc}}[\![B]\!] \ \wedge \ \{n\} \not\sqsubseteq \pi_j.M^t_{\mathit{msc}}[\![A]\!] \ \wedge \ \varGamma^t.n \preceq [\epsilon]$:

$$\varGamma^t.((\|i : i \neq j \,:\, \pi_i.g_i) \ \| \ n) \ \preceq \ \varGamma^t.(\|i : i \neq j \,:\, \pi_i.g_i)$$

– $\forall \ h :: [I \rightarrow M^t_{\mathit{msc}}[\![B]\!]],\ m :\ m \in \pi_j.M^t_{\mathit{msc}}[\![A]\!] \wedge \{m\} \not\sqsubseteq \pi_j.M^t_{\mathit{msc}}[\![B]\!] \wedge \varGamma^t.m \preceq [\epsilon]$:

$$\varGamma^t.((\|i : i \neq j \,:\, \pi_i.h_i) \ \| \ m) \ \preceq \ \varGamma^t.(\|i : i \neq j \,:\, \pi_i.h_i)$$

### 7.3 Non-deterministic choice

Propagating choice is an important property, but it is not easy to apply. A simple case that violates it is when the MSCs contain behaviors $m$ and $n$ that are different, although they share a common prefix $p$, i.e. $p \preceq m$ and $p \preceq n$. In case such a prefix $p$ starts with a receipt behavior, instance $j$ cannot resolve the choice using one of its initial events. This is characterized by the *non-deterministic choice* criterion (see also [MGR05]):

$$\begin{array}{l}(\exists j,m,n,p ::\ p \preceq m \ \wedge \ p \preceq n \,\wedge \\ \qquad\qquad\qquad m \in \pi_j.M^t_{\mathit{msc}}[\![A]\!] \ \wedge \ \{m\} \not\sqsubseteq \pi_j.M^t_{\mathit{msc}}[\![B]\!] \ \wedge \ \varGamma^t.m \preceq [\epsilon] \\ \qquad\qquad\quad \wedge\, n \in \pi_j.M^t_{\mathit{msc}}[\![B]\!] \ \wedge \ \{n\} \not\sqsubseteq \pi_j.M^t_{\mathit{msc}}[\![A]\!] \ \wedge \ \varGamma^t.n \preceq [\epsilon] \\ \qquad\qquad\quad \wedge\, (\exists g,h :\ g :: [I \rightarrow M^t_{\mathit{msc}}[\![A]\!]] \ \wedge \ h :: [I \rightarrow M^t_{\mathit{msc}}[\![B]\!]] : \\ \qquad\qquad\qquad\quad (\varGamma^t.((\|i : i \neq j \,:\, \pi_i.g_i) \ \| \ p) \ \not\preceq \ \varGamma^t.(\|i : i \neq j \,:\, \pi_i.g_i) \\ \qquad\qquad\qquad\qquad \vee\, \varGamma^t.((\|i : i \neq j \,:\, \pi_i.h_i) \ \| \ p) \ \not\preceq \ \varGamma^t.(\|i : i \neq j \,:\, \pi_i.h_i)) \ ) \ )\end{array}$$

This criterion can be made more syntactic by weakening the inner existential quantification into condition $p \not\preceq [\epsilon]$. Although non-deterministic choice violates safe choice, it does not guarantee that the derived condition in Section 6 is violated; so safe choice has been a real strengthening.

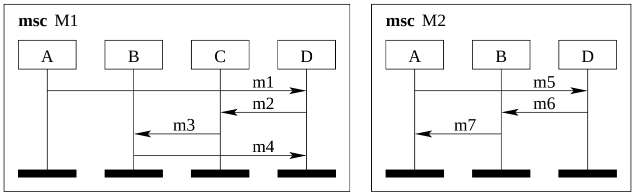


**Fig. 2.** Example from [HJ00]

### 7.4 Race choice

Absence of non-deterministic choice is not sufficient to guarantee propagating choice. It does guarantee that each instance $j$ can resolve the choice when no initiating receipt event can end up receiving a message intended for a non-initial receipt event in another MSC. The other cases are characterized by the *race choice* criterion (see also [MGR05], compare race conditions):

$$
\begin{aligned}
(\exists j :: \; & (\exists g, n :: \; g :: [I \rightarrow M^t_{msc}\llbracket A \rrbracket] \\
& \qquad \wedge n \in \pi_j.M^t_{msc}\llbracket B \rrbracket \;\wedge\; \{n\} \not\sqsubseteq \pi_j.M^t_{msc}\llbracket A \rrbracket \;\wedge\; \varGamma^t.n \preceq [\epsilon] \\
& \qquad \wedge \varGamma^t.((\|i : i \neq j \, : \, \pi_i.g_i) \; \| \; n) \;\; \not\preceq \;\; \varGamma^t.(\|i : i \neq j \, : \, \pi_i.g_i) \\
& \qquad \wedge (\forall p : p \preceq n \;\wedge\; \{p\} \sqsubseteq \pi_j.M^t_{msc}\llbracket A \rrbracket \; : \\
& \qquad\qquad \varGamma^t.((\|i : i \neq j \, : \, \pi_i.g_i) \; \| \; p) \;\; \preceq \;\; \varGamma^t.(\|i : i \neq j \, : \, \pi_i.g_i) \, ) \, ) \\
& \vee (\exists h, m :: \; h :: [I \rightarrow M^t_{msc}\llbracket B \rrbracket] \\
& \qquad \wedge m \in \pi_j.M^t_{msc}\llbracket A \rrbracket \;\wedge\; \{m\} \not\sqsubseteq \pi_j.M^t_{msc}\llbracket B \rrbracket \;\wedge\; \varGamma^t.n \preceq [\epsilon] \\
& \qquad \wedge \varGamma^t.((\|i : i \neq j \, : \, \pi_i.h_i) \; \| \; m) \;\; \not\preceq \;\; \varGamma^t.(\|i : i \neq j \, : \, \pi_i.h_i) \\
& \qquad \wedge (\forall p : p \preceq m \;\wedge\; \{p\} \sqsubseteq \pi_j.M^t_{msc}\llbracket B \rrbracket \; : \\
& \qquad\qquad \varGamma^t.((\|i : i \neq j \, : \, \pi_i.h_i) \; \| \; p) \;\; \preceq \;\; \varGamma^t.(\|i : i \neq j \, : \, \pi_i.h_i) \, ) \, ) \, )
\end{aligned}
$$

In [HJ00] the reconstructible choice criterion is proposed in order to guarantee realizability, and it is mentioned explicitly that the communication channels are not assumed to be order preserving. However, this claim contradicts their example of a reconstructible MSC [HJ00, Figure 15].

To illustrate our race choice criterion, we have copied the bMSCs from this example into Figure 2. The high-level MSC (which contains iteration) can be characterized as the smallest solution of:

$$M : \; M \; = \; (M_1 \, \mathbf{seq} \, M) \;\; \mathbf{alt} \;\; (M_2 \, \mathbf{seq} \, M)$$

Implementations allow behaviors that start as depicted in Figure 3, but prefix $!(m_1, A, D) \cdot !(m_5, A, D) \cdot ?(m_5, A, D)$ shows that this behavior is not part of the specified behavior.

In terms of our classification, this example suffers from race choice. Possible witnesses of the existential quantifications in its definition are characterized by

$j : \; j = D$<br>
$n : \; \pi_j.M_{msc}\llbracket M_2 \, \mathbf{seq} \, M_1 \rrbracket \; \preceq \; n$<br>
$g : \; (\forall i : i \neq j \, : \; M_{msc}\llbracket M_1 \, \mathbf{seq} \, M_2 \rrbracket \; \preceq \; g.i)$

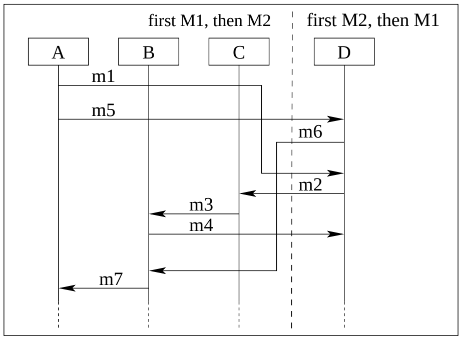


**Fig. 3.** Execution behavior of the example from [HJ00]

## 8 Conclusions and further work

We have developed a denotational semantics for compositional MSC through our extension of pomsets with deadlocks. In this formalism we have studied realizability, especially of the choice construct. We have discussed various proposed realizability criteria and shown completeness of our classification in [MGR05].

Realizability problems can also be detected by verifying the implementation [UKM03]. However, it is far more effective to have criteria for specifications, and to develop ways to make specifications realizable [HJ00]. For the latter, we plan to evaluate our proposals in [MG05,MGR05] using the current framework, and to automate them.

A possible extension is to explore other realizability criteria, especially since safe choice is a real strengthening. In addition, more syntactical criteria would better allow automation. Also the realizability of other MSC constructs may be studied, of which parallel composition is a challenging one.

## References

[BAL97] H. Ben-Abdallah and S. Leue. Syntactic detection of process divergence and non-local choice in Message Sequence Charts. In *Proceedings of TACAS'97*, volume 1217 of *LNCS*, pages 259–274. Springer, 1997.

[BM95] J.C.M. Baeten and S. Mauw. Delayed choice: an operator for joining Message Sequence Charts. In *Formal Description Techniques*, pages 340–354, 1995.

[Gen05] B. Genest. Compositional Message Sequence Charts (CMSCs) are better to implement than MSCs. In *Proceedings of TACAS'05*, volume 3440 of *LNCS*, pages 429–440. Springer, 2005.

[GMP03] E.L. Gunter, A. Muscholl, and D.A. Peled. Compositional Message Sequence Charts. *International Journal on Software Tools for Technology Transfer*, 5(1):78–89, November 2003. An earlier version appeared at TACAS'01.

[Hey00] S. Heymer. A semantics for MSC based on Petri-Net components. In *Proceedings of SAM'00: 2nd Workshop on SDL and MSC*, 2000.

[HJ00] L. Hélouët and C. Jard. Conditions for synthesis of communicating automata from HMSCs. In *Proceedings of 5th FMICS Workshop*, 2000.

[KL98] J.-P. Katoen and L. Lambert. Pomsets for Message Sequence Charts. In *Proceedings of SAM'98: 1st Workshop on SDL and MSC*, 1998.

[MG05] A.J. Mooij and N. Goga. Dealing with non-local choice in IEEE 1073.2's standard for remote control. In *Proceedings of SAM'04: 4th Workshop on SDL and MSC*, volume 3319 of *LNCS*, pages 257–270. Springer, 2005.

[MGR05] A.J. Mooij, N. Goga, and J.M.T. Romijn. Non-local choice and beyond: Intricacies of MSC choice nodes. In *Proceedings of FASE'05*, volume 3442 of *LNCS*, pages 273–288. Springer, 2005.

[MM01] P. Madhusudan and B. Meenakshi. Beyond message sequence graphs. In *Proceedings of FASE'01*, LNCS 2245, pages 256–267. Springer, 2001.

[Pra86] V. Pratt. Modelling concurrency with partial orders. *International Journal of Parallel Programming*, 15(1):33–71, 1986.

[Ren99] M.A. Reniers. *Message Sequence Chart: Syntax and Semantics*. PhD thesis, Technische Universiteit Eindhoven, June 1999.

[UKM03] S. Uchitel, J. Kramer, and J. Magee. Synthesis of behavioral models from scenarios. *IEEE Transactions on Software Engineering*, 29(2):99–115, 2003.

# A Proofs about the prefix order on laters

## A.1 Corollary of the definition

We first prove that $\overline{E_p} \subseteq \overline{E_q}$ is a corollary of $(E_p, <_p, l_p) \preceq (E_q, <_q, l_q)$.

$\quad \overline{E_p} \subseteq \overline{E_q}$

$\equiv \quad \{$set calculus; definition of $\overline{E_q}\}$

$\quad (\forall f : f \in \overline{E_p} : f \in E_q \wedge f \notin \mathit{\Delta}.(E_q, <_q, l_q))$

$\equiv \quad \{$definition of $\mathit{\Delta}\}$

$\quad (\forall f : f \in \overline{E_p} : f \in E_q \wedge (\forall e : e <_q e : \neg(e <_q f)))$

$\Leftarrow \quad \{$condition 1; condition 4: $f \in \overline{E_p} \wedge e \notin \overline{E_p} \Rightarrow \neg(e <_q f)\}$

$\quad (\forall e : e <_q e : e \notin \overline{E_p})$

$\equiv \quad \{$proof by contradiction; definition of $\overline{E_p}\}$

$\quad (\forall e : e <_q e \wedge e \in \overline{E_p} : e \in \mathit{\Delta}(E_p, <_p, l_q))$

$\equiv \quad \{$condition 2 gives $e <_p e$; definition of $\mathit{\Delta}\}$

$\quad \mathit{true}$

## A.2 Variants of the definition

To simplify some future proofs, we prove that exploiting condition 4, condition 2 is equivalent to the *stronger* condition $<_q \cap (E_p \times \overline{E_p}) \subseteq <_p$.

$\quad <_q \cap (E_p \times \overline{E_p}) \subseteq <_p$

$\Leftarrow \quad \{$condition 2$\}$

$\quad <_q \cap (E_p \times \overline{E_p}) \subseteq <_q \cap (\overline{E_p} \times \overline{E_p})$

$\equiv \quad \{$set calculus$\}$

$\quad <_q \cap (E_p \times \overline{E_p}) \subseteq (\overline{E_p} \times \overline{E_p})$

$\equiv \quad \{$set calculus; condition 4$\}$

$\quad \mathit{true}$

After strengthening condition 2, condition 4 is equivalent to the *weaker* condition $(\forall e, f : e <_q f \wedge f \in \overline{E_p} : e \in E_p)$. We prove it by showing how it can be used to prove condition 4:

$\quad (\forall e, f : e <_q f \wedge f \in \overline{E_p} : e \in \overline{E_p})$

$\equiv \quad \{$weak condition 4 gives $e \in E_p$; definition of $\overline{E_p}\}$

$\quad (\forall e, f : e <_q f \wedge f \in \overline{E_p} \wedge e \in E_p : e \notin \mathit{\Delta}.(E_p, <_p, l_p))$

$\Leftarrow \quad \{$strong condition 2$\}$

$\quad (\forall e, f : e <_p f \wedge f \in \overline{E_p} : e \notin \mathit{\Delta}.(E_p, <_p, l_p))$

$\equiv \quad \{$trading; definition of $\overline{E_p}\}$

$\quad (\forall e, f : e <_p f \wedge e \in \mathit{\Delta}.(E_p, <_p, l_p) : f \in \mathit{\Delta}.(E_p, <_p, l_p))$

$\equiv \quad \{$definition of $\mathit{\Delta}$; transitivity of $<_p\}$

$\quad \mathit{true}$

If $E_q \subseteq E_p$ then weak condition 4 reduces to true.

### A.3 Transitivity

We prove transitivity of $\preceq$ by assuming that $(E_p, <_p, l_p) \preceq (E_q, <_q, l_q)$ and $(E_q, <_q, l_q) \preceq (E_r, <_r, l_r)$. Using the definition of $\preceq$ we thus have:

| | | | |
|---|---|---|---|
| 1pq: | $\overline{E_p} \subseteq E_q$ | 1qr: | $\overline{E_q} \subseteq E_r$ |
| 2pq: | $<_q \cap (\overline{E_p} \times \overline{E_p}) \subseteq <_p$ | 2qr: | $<_r \cap (\overline{E_q} \times \overline{E_q}) \subseteq <_q$ |
| 3pq: | $l_p \cap (\overline{E_p} \times L) = l_q \cap (\overline{E_p} \times L)$ | 3qr: | $l_q \cap (\overline{E_q} \times L) = l_r \cap (\overline{E_q} \times L)$ |
| 4pq: | $(\forall e, f : e <_q f \wedge f \in \overline{E_p} : e \in \overline{E_p})$ | 4qr: | $(\forall e, f : e <_r f \wedge f \in \overline{E_q} : e \in \overline{E_q})$ |

Then we show $(E_p, <_p, l_p) \preceq (E_r, <_r, l_r)$ by proving the four conjuncts corresponding to the definition of $\preceq$:

$\quad \overline{E_p} \subseteq E_r$
$\Leftarrow \quad \{1\text{qr}\}$
$\quad \overline{E_p} \subseteq \overline{E_q}$
$\equiv \quad \{\text{corollary } \overline{E_p} \subseteq \overline{E_q}\}$
$\quad \mathit{true}$

$\quad <_r \cap (\overline{E_p} \times \overline{E_p}) \subseteq <_p$
$\Leftarrow \quad \{2\text{pq}\}$
$\quad <_r \cap (\overline{E_p} \times \overline{E_p}) \subseteq <_q \cap (\overline{E_p} \times \overline{E_p})$
$\equiv \quad \{\text{set calculus}\}$
$\quad <_r \cap (\overline{E_p} \times \overline{E_p}) \subseteq <_q$
$\Leftarrow \quad \{2\text{qr}\}$
$\quad <_r \cap (\overline{E_p} \times \overline{E_p}) \subseteq <_r \cap (\overline{E_q} \times \overline{E_q})$
$\equiv \quad \{\text{set calculus}\}$
$\quad <_r \cap (\overline{E_p} \times \overline{E_p}) \subseteq (\overline{E_q} \times \overline{E_q})$
$\equiv \quad \{\text{corollary } \overline{E_p} \subseteq \overline{E_q}\}$
$\quad \mathit{true}$

$\quad l_p \cap (\overline{E_p} \times L) = l_r \cap (\overline{E_p} \times L)$
$\equiv \quad \{3\text{pq}\}$
$\quad l_q \cap (\overline{E_p} \times L) = l_r \cap (\overline{E_p} \times L)$
$\Leftarrow \quad \{\text{corollary } \overline{E_p} \subseteq \overline{E_q}\}$
$\quad l_q \cap (\overline{E_q} \times L) = l_r \cap (\overline{E_q} \times L)$
$\equiv \quad \{3\text{qr}\}$
$\quad \mathit{true}$

$\quad (\forall e, f : e <_r f \wedge f \in \overline{E_p} : e \in \overline{E_p})$
$\Leftarrow \quad \{4\text{pq}\}$
$\quad (\forall e, f : e <_r f \wedge f \in \overline{E_p} : e <_q f)$
$\Leftarrow \quad \{\text{corollary } \overline{E_p} \subseteq \overline{E_q}\}$
$\quad (\forall e, f : e <_r f \wedge f \in \overline{E_q} : e <_q f)$
$\Leftarrow \quad \{2\text{qr}\}$
$\quad (\forall e, f : e <_r f \wedge f \in \overline{E_q} : e \in \overline{E_q})$
$\equiv \quad \{4\text{qr}\}$
$\quad \mathit{true}$

### A.4 Monotonicity with respect to both sequential compositions

Let $m = (E_m, <_m, l_m)$, $p = (E_p, <_p, l_p)$ and $m \circ_W p = (E_{mp}, <_{mp}, l_{mp})$, where $E_{mp} = E_m \cup E_p$, $<_{mp} = (<_m \cup <_{\circ_{mp}} \cup <_p)^+$ and $l_{mp} = l_m \cup l_p$. We assume that the event sets are such that $E_m \cap E_p = \emptyset$ and $E_m \cap E_q = \emptyset$. To eliminate the transitive closure in the definition of $<_{mp}$, we can use that the event sets of $E_m$ and $E_p$ are disjoint, $<_m$ and $<_p$ are transitive, and $<_{\circ_{mp}} \subseteq E_m \times E_p$. Thus $d <_{mp} g$ is equivalent to:

$$d <_m g \;\vee\; d <_p g \;\vee\; (\exists e, f :: \; (d <_m e \vee d = e) \;\wedge\; e <_{\circ_{mp}} f \;\wedge\; (f = g \vee f <_p g))$$

Assuming $p \preceq q$, we show $m \circ p \;\preceq\; m \circ q$ by proving the four conjuncts of the definition of $\preceq$ (strong second, weak fourth). We will use that $\overline{E_{mp}} \;\subseteq\; \overline{E_m} \cup \overline{E_p}$ holds since $\circ$ only adds causalities.

$$\begin{array}{ll}
 & \overline{E_{mp}} \;\subseteq\; E_{mq} \\
\Leftarrow & \quad \{\overline{E_{mp}} \;\subseteq\; \overline{E_m} \cup \overline{E_p}\}\{E_{mq} = E_m \cup E_q\} \\
 & (\overline{E_m} \cup \overline{E_p}) \;\subseteq\; E_m \cup E_q \\
\equiv & \quad \{\text{by definition } \overline{E_m} \subseteq E_m\}\{\text{condition 1: } \overline{E_p} \subseteq E_q\} \\
 & \mathit{true}
\end{array}$$

$$\begin{array}{ll}
 & d <_{mp} g \\
\equiv & \quad \{\text{definition of } <_{mp}\} \\
 & d <_m g \;\vee\; d <_p g \vee \\
 & \qquad (\exists e, f :: \; (d <_m e \vee d = e) \;\wedge\; e <_{\circ_{mp}} f \;\wedge\; (f = g \vee f <_p g)) \\
\Leftarrow & \quad \{\text{strong condition 2}\} \\
 & d <_m g \;\vee\; (g \in \overline{E_p} \;\wedge\; ((d \in E_p \;\wedge\; d <_q g) \vee \\
 & \qquad (\exists e, f :: \; (d <_m e \vee d = e) \;\wedge\; e <_{\circ_{mp}} f \;\wedge\; (f = g \vee f <_q g))\;)) \\
\Leftarrow & \quad \{\text{condition 4: } f \in \overline{E_p}\}\{\text{property of } \circ\text{, use condition 3: } l_p.f = l_q.f\} \\
 & d <_m g \;\vee\; (g \in \overline{E_p} \;\wedge\; ((d \in E_p \;\wedge\; d <_q g) \vee \\
 & \qquad (\exists e, f :: \; (d <_m e \vee d = e) \;\wedge\; e <_{\circ_{mq}} f \;\wedge\; (f = g \vee f <_q g))\;)) \\
\Leftarrow & \quad \{E_{mp} = E_m \cup E_p\}\{\overline{E_{mp}} \;\subseteq\; \overline{E_m} \cup \overline{E_p}\}\{\text{use } E_m \cap E_q = \emptyset\} \\
 & d \in E_{mp} \;\wedge\; g \in \overline{E_{mp}} \;\wedge\; (d <_m g \;\vee\; d <_q g \vee \\
 & \qquad (\exists e, f :: \; (d <_m e \vee d = e) \;\wedge\; e <_{\circ_{mq}} f \;\wedge\; (f = g \vee f <_q g))\;) \\
\equiv & \quad \{\text{definition of } <_{mq}\} \\
 & d \in E_{mp} \;\wedge\; g \in \overline{E_{mp}} \;\wedge\; d <_{mq} g
\end{array}$$

$$\begin{array}{ll}
 & l_{mp} \cap (\overline{E_{mp}} \times L) \;=\; l_{mq} \cap (\overline{E_{mp}} \times L) \\
\equiv & \quad \{l_{mp} = l_m \cup l_p\}\{l_{mq} = l_m \cup l_q\}\{\text{set calculus}\} \\
 & l_p \cap (\overline{E_{mp}} \times L) \;=\; l_q \cap (\overline{E_{mp}} \times L) \\
\Leftarrow & \quad \{\overline{E_{mp}} \;\subseteq\; \overline{E_m} \cup \overline{E_p}\}\{E_m \cap E_p = \emptyset\}\;\{E_m \cap E_q = \emptyset\} \\
 & l_p \cap (\overline{E_p} \times L) \;=\; l_q \cap (\overline{E_p} \times L) \\
\equiv & \quad \{\text{condition 3}\} \\
 & \mathit{true}
\end{array}$$

$$
\begin{array}{ll}
 & d \in E_{mp} \\
\equiv & \quad \{\text{definition of } E_{mp}\} \\
 & d \in E_m \ \vee \ d \in E_p \\
\Leftarrow & \quad \{\text{weak condition 4}\} \\
 & d \in E_m \ \vee \ (\exists g :: \ g \in \overline{E_p} \ \wedge \ d <_q g) \\
\Leftarrow & \quad \{\overline{E_{mp}} \ \subseteq \ \overline{E_m} \cup \overline{E_p}, \ \overline{E_m} \subseteq E_m, \text{ and } E_m \cap E_q = \emptyset\} \\
 & (\exists g :: \ g \in \overline{E_{mp}} \ \wedge \ (d <_m g \ \vee \ d <_q g \vee \\
 & \qquad (\exists e, f :: \ (d <_m e \vee d = e) \ \wedge \ e <_{\circ_{mq}} f \ \wedge \ (f = g \vee f <_q g)) \,)) \\
\equiv & \quad \{\text{definition of } <_{mq}\} \\
 & (\exists g :: \ g \in \overline{E_{mp}} \ \wedge \ d <_{mq} g)
\end{array}
$$

# B Proofs about communication operator $\mathit{\Gamma}$

## B.1 Idempotence

Let $p = (E, <, l)$, $\Gamma^t.p = (E', <', l')$ and $\Gamma^t.(\Gamma^t.p) = (E'', <'', l'')$. Since $\Gamma$ is a causality extension, $E' = E''$, $<' \subseteq <''$ and $l' = l''$, and hence we only need to prove $<'' \subseteq <'$ to show that $\Gamma$ is idempotent.

Using the label-wise trichotomy properties of $t$ and $p$, all causalities that are added via $<''_c$ are already present in $<'_c$. For the causalities in $<''_d$ we must consider a receipt event that has a matching send event in $<'_c$ but not in $<''_c$. In $\Gamma^t.p$ this send event is behind a deadlock, and hence also this receipt event is behind a deadlock. Hence this receipt event is not in $<''_d$. So $<' \subseteq <$ is guaranteed.

## B.2 Monotonicity

Assuming $p \preceq q$, we will prove $\Gamma^t.p \preceq \Gamma^t.q$ by considering the four conditions for $\preceq$. Let $p = (E_p, <_p, l_p)$, $q = (E_q, <_q, l_q)$, $\Gamma^t.p = (E_p^\gamma, <_p^\gamma, l_p^\gamma)$ and $\Gamma^t.q = (E_q^\gamma, <_q^\gamma, l_q^\gamma)$. Since $\Gamma^t$ is a causality extension, we have $\overline{E_p^\gamma} \subseteq \overline{E_p}$ and $E_q = E_q^\gamma$. This observation completes the proof of conditions 1 and 3. What remains are strong condition 2 and weak condition 4. Since they are maintained under shrinking $\overline{E_p}$ to $\overline{E_p^\gamma}$ and extending $<_p$ to $<_p^\gamma$, we only need to consider an order $d <_q^\gamma g$ for $g \in \overline{E_p^\gamma}$ while $\neg(d <_q g)$. We consider the two extensions:

- adding $<_{q_c}$ and applying the transitive closure: then there exists an interleaving of steps from $<_q$ and $<_{q_c}$ that witnesses $d <_q^\gamma g$. Thanks to strong condition 2, each step $e <_q f$ for $f \in \overline{E_p^\gamma}$ (and hence $f \in \overline{E_p}$) guarantees $e <_p f$, and hence by definition we have $e \in \overline{E_p^\gamma}$. Since $p \preceq q$, each step $e <_{q_c} f$ for $f \in \overline{E_p^\gamma}$ guarantees $e <_{p_c} f$ and hence by definition we have $e \in \overline{E_p^\gamma}$. Hence we can conclude $d <_p^\gamma g$, which establishes strong condition 2 and weak condition 4.
- adding $<_{q_d}$: then $d = g$ and weak condition 4 clearly holds. Since $g$ is a receipt event, $g \in \overline{E_p^\gamma}$ and $p \preceq q$, also $d <_p^\gamma g$ is added, which establishes strong condition 2.

### B.3 Property regarding sequential composition

We split the proof of $\doteq$ in its two directions:

$\quad \varGamma^t.(\{p\} \circ_W \varGamma^{t \circ_W p}.Q) \;\sqsubseteq\; \varGamma^t.(\{p\} \circ_W Q)$
$\Leftarrow \quad \{\text{monotonicity of } \varGamma\}$
$\quad \{p\} \circ_W \varGamma^{t \circ_W p}.Q \;\sqsubseteq\; \{p\} \circ_W Q$
$\Leftarrow \quad \{\text{monotonicity of } \circ_W\}$
$\quad \varGamma^{t \circ_W p}.Q \;\sqsubseteq\; Q$
$\equiv \quad \{\text{shrinking } \varGamma\}$
$\quad \mathit{true}$

$\quad \varGamma^t.(\{p\} \circ_W Q) \;\sqsubseteq\; \varGamma^t.(\{p\} \circ_W \varGamma^{t \circ_W p}.Q)$
$\equiv \quad \{\text{idempotence of } \varGamma\}$
$\quad \varGamma^t.(\varGamma^t.(\{p\} \circ_W Q)) \;\sqsubseteq\; \varGamma^t.(\{p\} \circ_W \varGamma^{t \circ_W p}.Q)$
$\Leftarrow \quad \{\text{monotonicity of } \varGamma\}$
$\quad \varGamma^t.(\{p\} \circ_W Q) \;\sqsubseteq\; \{p\} \circ_W \varGamma^{t \circ_W p}.Q$
$\Leftarrow \quad \{\text{calculus}\}$
$\quad (\forall q : q \in Q : \; \varGamma^t.(p \circ_W q) \;\preceq\; p \circ_W \varGamma^{t \circ_W p}.q)$

For the remaining $\preceq$, note that the event sets and the labeling are identical, and hence we only need to consider strong condition 2. Since $\circ_W$ is associative, $(E', <', l')$ is identical in both $\varGamma$'s. Since the events of $q$ are contained in the events of $p \circ_W q$, the orders introduced by $\varGamma$ in the right term are a subset of the orders introduced by $\varGamma$ in the left term.

### B.4 Deadlock extension rule

Provided laters $x$ and $y$ are label disjoint and $y = (E_y, <_y, l_y)$:

$$E_y \subseteq \varDelta.(\varGamma^t.(x \| y)) \;\equiv\; \varGamma^t.(x \| y) \;\preceq\; \varGamma^t.x$$

$\Leftarrow$ follows from condition 1 of $\preceq$. For $\Rightarrow$ we consider the four conditions for $\preceq$. Condition 1 is guaranteed, and hence also condition 3 is guaranteed. Weak condition 4 is guaranteed since the events in $x$ are contained in the events in $x \| y$.

For strong condition 2 we need to show that each causality $a < b$ from $\varGamma^t.x$ such that $b \notin \varDelta.(\varGamma^t.(x \| y))$ is also in $\varGamma^t.(x \| y)$. This holds trivially for the causalities from $x$. Thanks to label-disjointness of $x$ and $y$, it holds for the causalities that are introduced via $<_c$. Finally, it holds for the causalities that are introduced via $<_d$ by using $b \notin \varDelta.(\varGamma^t.(x \| y))$ and $E_y \subseteq \varDelta.(\varGamma^t.(x \| y))$.

### B.5 Multiple deadlock extension rule

Provided laters $x$, $y$ and $z$ are label disjoint:

$$\varGamma^t.(x \| y) \;\preceq\; \varGamma^t.x \;\wedge\; \varGamma^t.(x \| z) \;\preceq\; \varGamma^t.x \quad\equiv\quad \varGamma^t.(x \| y \| z) \;\preceq\; \varGamma^t.x$$

$\Leftarrow$ follows from monotonicity. For $\Rightarrow$ we can use the deadlock extension rule by showing that all events from $y\|z$ are in $\Delta.(\Gamma^t.(x\|y\|z))$. Applying the deadlock extension rule to the left-hand side gives that the events from $y$ and $z$ are in $\Delta.(\Gamma^t.(x\|y))$ and $\Delta.(\Gamma^t.(x\|z))$ respectively. Hence all possibly first events in $y$ and $z$ are receipts that are not provided by $\Gamma^t.x$ alone. This ensures that all events from $y$ and $z$ are in $\Delta.(\Gamma^t.(x\|y\|z))$.

### B.6 Elimination rule

Provided laters $x$, $y$ and $z$ are label disjoint:

$$\Gamma^t.(x\|y\|z) \preceq \Gamma^t.(x\|y) \;\Rightarrow\; \Gamma^t.(x\|z) \preceq \Gamma^t.x$$

Using the deadlock extension rule, it is sufficient to show that all events from $z$ are in $\Delta.(\Gamma^t.(x\|z))$. Applying the deadlock extension rule to the antecedent gives that the events from $z$ are in $\Delta.(\Gamma^t.(x\|y\|z))$. Hence all possibly first events in $z$ are receipts that are not provided by $\Gamma^t.(x\|y)$ alone. Since $\Gamma^t.x \preceq \Gamma^t.(x\|y)$, all events from $z$ are in $\Delta.(\Gamma^t.(x\|z))$.

As a corollary ($x := [\epsilon]$) we have $\Gamma^t.(y\|z) \preceq \Gamma^t.y \Rightarrow \Gamma^t.z \preceq [\epsilon]$.

## C Proofs about implementations

### C.1 Monotonicity of $(R^t \circ D)$ with respect to causality extensions

We prove:

$$\begin{array}{ll}
 & (R^t \circ D).(\Gamma^t.M) \sqsubseteq (R^t \circ D).M \\
\equiv & \{\text{definition of } R^t \circ D\} \\
 & \Gamma^t.(\|i : i \in I : \pi_i.(\Gamma^t.M)) \sqsubseteq \Gamma^t.(\|i : i \in I : \pi_i.M) \\
\Leftarrow & \{\text{monotonicity of } \Gamma\} \\
 & (\|i : i \in I : \pi_i.(\Gamma^t.M)) \sqsubseteq (\|i : i \in I : \pi_i.M) \\
\Leftarrow & \{\text{property of } \|\} \\
 & (\forall i : i \in I : \pi_i.(\Gamma^t.M) \sqsubseteq \pi_i.M) \\
\Leftarrow & \{\text{calculus}\} \\
 & (\forall i, m : i \in I \wedge m \in M : \pi_i.(\Gamma^t.m) \preceq \pi_i.m) \\
\equiv & \{\text{property of } \preceq, \pi \text{ and causality extension } \Gamma^t\} \\
 & \mathit{true}
\end{array}$$

### C.2 Distribution of $\circ_W$ over $(R^t \circ D)$

For $m$ a later that does not order events in different instances, and $M$ a set of laters, we prove:

$$
\begin{array}{ll}
 & (R^t \circ D).(\{m\} \circ_W M) \\
= & \quad \{\text{definition of } R \circ D\} \\
 & \mathit{\Gamma}^t.(\|i : i \in I :\ \pi_i.(\{m\} \circ_W M)) \\
= & \quad \{\text{distribution}\} \\
 & \mathit{\Gamma}^t.(\|i : i \in I :\ \pi_i.\{m\}\ \circ_W\ \pi_i.M) \\
= & \quad \{\text{distribution, since } m \text{ does not order events in different instances}\} \\
 & \mathit{\Gamma}^t.(\{m\} \circ_W (\|i : i \in I :\ \pi_i.M)) \\
\doteq & \quad \{\text{property of } \mathit{\Gamma} \text{ and } \circ_W\} \\
 & \mathit{\Gamma}^t.(\{m\} \circ_W \mathit{\Gamma}^{t \circ_W m}.(\|i : i \in I :\ \pi_i.M)) \\
= & \quad \{\text{definition of } R \circ D\} \\
 & \mathit{\Gamma}^t.(\{m\} \circ_W (R^{t \circ_W m} \circ D).M)
\end{array}
$$

This proof uses that sequential composition is *weak*. In view of the graphical syntax of MSC, it would be more natural to define sequential composition as strong. However, the above rule only holds for weak sequential composition. If we would start from the top of the above proof to replace $\circ_W$ by $\circ_S$, then after the third step we get stuck and need $\circ_W$ again. Although this does not prove that strong sequential composition is infeasible, it is at least an indication that weak sequential composition might be the strongest one that is realizable.

### C.3 Safe choice

We simplify and strengthen the derived condition for choice in two steps. We first concentrate on the first disjunct:

$$
\begin{array}{ll}
 & \mathit{\Gamma}^t.(\|i ::\ \pi_i.f_i)\ \preceq\ \mathit{\Gamma}^t.(\|i ::\ \pi_i.g_i) \\
\Leftarrow & \quad \{\text{ monotonicity }\} \\
 & \mathit{\Gamma}^t.(\|i ::\ \pi_i.f_i)\ \preceq\ \mathit{\Gamma}^t.(\|i : \pi_i.f_i \preceq \pi_i.g_i :\ \pi_i.g_i) \\
\Leftarrow & \quad \{\text{ domain split; monotonicity }\} \\
 & \mathit{\Gamma}^t.((\|i : \pi_i.f_i \preceq \pi_i.g_i :\ \pi_i.g_i)\ \|\ (\|i : \pi_i.f_i \not\preceq \pi_i.g_i :\ \pi_i.f_i)) \\
 & \qquad \preceq\ \mathit{\Gamma}^t.(\|i : \pi_i.f_i \preceq \pi_i.g_i :\ \pi_i.g_i) \\
\equiv & \quad \{\text{ property of } \mathit{\Gamma} \text{ (multiple deadlock extension rule) }\} \\
 & (\forall j : \pi_j.f_j \not\preceq \pi_j.g_j : \\
 & \quad \mathit{\Gamma}^t.((\|i : \pi_i.f_i \preceq \pi_i.g_i :\ \pi_i.g_i)\ \|\ \pi_j.f_j)\ \preceq\ \mathit{\Gamma}^t.(\|i : \pi_i.f_i \preceq \pi_i.g_i :\ \pi_i.g_i)\ ) \\
\Leftarrow & \quad \{\text{ property of } \mathit{\Gamma} \text{ (elimination rule) }\} \\
 & (\forall j : \pi_j.f_j \not\preceq \pi_j.g_j : \\
 & \quad \mathit{\Gamma}^t.((\|i : i \neq j :\ \pi_i.g_i)\ \|\ \pi_j.f_j)\ \preceq\ \mathit{\Gamma}^t.(\|i : i \neq j :\ \pi_i.g_i)\ )
\end{array}
$$

Let us abbreviate $\mathit{\Gamma}^t.((\|i : i \neq j : \pi_i.g_i) \| \pi_j.f_j)\ \preceq\ \mathit{\Gamma}^t.(\|i : i \neq j : \pi_i.g_i)$ as $P.g.j.f_j$. Then we can prove the remainder as follows:

$$
\begin{array}{ll}
 & (\forall f :: \\
 & \quad (\exists g :: \ (\forall j : \pi_j.f_j \not\preceq \pi_j.g_j : \ P.g.j.f_j)) \vee \\
 & \quad (\exists h :: \ (\forall j : \pi_j.f_j \not\preceq \pi_j.h_j : \ P.h.j.f_j)) \ ) \\
\Leftarrow & \quad \{ \text{ strengthening for later use } \} \\
 & (\forall f :: \ (\exists k :: \\
 & \quad (\exists g :: \ \pi_k.f_k \preceq \pi_k.g_k \ \wedge \ (\forall j : \pi_j.f_j \not\preceq \pi_j.g_j : \ P.g.j.f_j)) \vee \\
 & \quad (\exists h :: \ \pi_k.f_k \preceq \pi_k.h_k \ \wedge \ (\forall j : \pi_j.f_j \not\preceq \pi_j.h_j : \ P.h.j.f_j)) \ )) \\
\equiv & \quad \{ \text{ case } j = k \text{ follows from left conjunct } \} \\
 & (\forall f :: \ (\exists k :: \\
 & \quad (\exists g :: \ \pi_k.f_k \preceq \pi_k.g_k \ \wedge \ (\forall j : \pi_j.f_j \not\preceq \pi_j.g_j \wedge j \neq k : \ P.g.j.f_j)) \vee \\
 & \quad (\exists h :: \ \pi_k.f_k \preceq \pi_k.h_k \ \wedge \ (\forall j : \pi_j.f_j \not\preceq \pi_j.h_j \wedge j \neq k : \ P.h.j.f_j)) \ )) \\
\Leftarrow & \quad \{ \text{ use } (\forall f, k :: \\
 & \qquad (\exists g :: \ \pi_k.f_k \preceq \pi_k.g_k \ \wedge \ (\forall j : \pi_j.f_j \not\preceq \pi_j.g_j : \ \{\pi_j.f_j\} \not\sqsubseteq \pi_j.M^t_{msc}[\![A]\!])) \vee \\
 & \qquad (\exists h :: \ \pi_k.f_k \preceq \pi_k.h_k \ \wedge \ (\forall j : \pi_j.f_j \not\preceq \pi_j.h_j : \ \{\pi_j.f_j\} \not\sqsubseteq \pi_j.M^t_{msc}[\![B]\!])) \ ) \ \} \\
 & (\forall f :: \ (\exists k :: \\
 & \quad (\forall g, j : \{\pi_j.f_j\} \not\sqsubseteq \pi_j.M^t_{msc}[\![A]\!] \ \wedge \ j \neq k : \ P.g.j.f_j) \wedge \\
 & \quad (\forall h, j : \{\pi_j.f_j\} \not\sqsubseteq \pi_j.M^t_{msc}[\![B]\!] \ \wedge \ j \neq k : \ P.h.j.f_j) \ )) \\
\Leftarrow & \quad \{ \text{ quantifier shunting } \} \\
 & (\exists k :: \ (\forall j : j \neq k : \\
 & \quad (\forall f, g : \{\pi_j.f_j\} \not\sqsubseteq \pi_j.M^t_{msc}[\![A]\!] : \ P.g.j.f_j) \wedge \\
 & \quad (\forall f, h : \{\pi_j.f_j\} \not\sqsubseteq \pi_j.M^t_{msc}[\![B]\!] : \ P.h.j.f_j) \ )) \\
\equiv & \quad \{ \text{ dummy renaming } \} \\
 & (\exists k :: \ (\forall j : j \neq k : \\
 & \quad (\forall g, n : n \in \pi_j.M^t_{msc}[\![B]\!] \wedge \{n\} \not\sqsubseteq \pi_j.M^t_{msc}[\![A]\!] : \ P.g.j.n) \wedge \\
 & \quad (\forall h, m : m \in \pi_j.M^t_{msc}[\![A]\!] \wedge \{m\} \not\sqsubseteq \pi_j.M^t_{msc}[\![B]\!] : \ P.h.j.m) \ ))
\end{array}
$$