Operational Semantics of UML 2.0 Interactions

María Victoria Cengarle[1] and Alexander Knapp[2]

[1] Technische Universität München
cengarle@in.tum.de
[2] Ludwig-Maximilians-Universität München
knapp@pst.ifi.lmu.de

Abstract. An operational semantics for UML 2.0 Interactions is defined that for finite traces is compliant with the trace-based denotational semantics of a previous work. To this end, the notion of interactions in the classical sense is used. That is, the operational semantics of UML 2.0 Interactions is given by composing their translation into interactions in the classical sense and the reduction relations below.

1 Preliminaries

We briefly review the basic definitions on partially ordered, labelled multisets as introduced by Pratt [3] for modelling concurrency. In particular, we define sequential and parallel composition operators and the notion of traces and processes.

A partially ordered, labelled multiset, or *pomset*, is the isomorphism class $[(X, \leq_X, \lambda_X)]$ of a labelled partial order $(X, \leq_X, \lambda_X)$ w.r.t. monotone, label-preserving maps. A *trace* is a pomset whose ordering is total. We write $p{\downarrow}$ for all possible linearisations of a pomset p, i.e., all traces that extend the ordering of p: $[(X', \leq_{X'}, \lambda_{X'})] \in [(X, \leq_X, \lambda_X)]{\downarrow}$ if, and only if $X' = X$, $\lambda_{X'} = \lambda_X$, and $\leq_X \subseteq \leq_{X'}$ where $x_1 \leq_{X'} x_2$ or $x_2 \leq_{X'} x_1$ for all $x_1, x_2 \in X'$.

The *empty* pomset, represented by $(\emptyset, \emptyset, \emptyset)$, is denoted by ε. Let $p = [(X, \leq_X, \lambda_X)]$ and $q = [(Y, \leq_Y, \lambda_Y)]$ be pomsets such that $X \cap Y = \emptyset$. The *concurrence* of p and q, written as $p \| q$, is given by $[(X \cup Y, \leq_X \cup \leq_Y, \lambda_X \cup \lambda_Y)]$. The *concatenation* of p and q, written as $p\,;q$, is given by $[(X \cup Y, (\leq_X \cup \leq_Y \cup (X \times Y))^*, \lambda_X \cup \lambda_Y)]$. Given a binary, symmetric relation $\lessgtr$ on labels, the $\lessgtr$*-concatenation* of p and q, written as $p\,;_{\lessgtr} q$, is given by $[(X \cup Y, (\leq_X \cup \leq_Y \cup \{(x, y) \in X \times Y \mid \lambda_X(x) \lessgtr \lambda_Y(y)\})^*, \lambda_X \cup \lambda_Y)]$. We write p^n for the n-fold iteration of pomset concatenation with n a natural number, i.e., $p^0 = \varepsilon$ and $p^{n+1} = p\,; p^n$. Furthermore we let p^* denote $\bigcup_{n \geq 0} p^n$. Analogously, we write $p^{n_{\lessgtr}}$ for the n-fold iteration of $\lessgtr$-concatenation with $p^{0_{\lessgtr}} = \varepsilon$ and $p^{(n+1)_{\lessgtr}} = p\,;_{\lessgtr} p^{n_{\lessgtr}}$. Note that concatenation and $\lessgtr$-concatenation are associative, and concurrence is associative and commutative.

A *process* is a set of pomsets. An n-ary function f on pomsets is lifted to processes $P_1, \ldots, P_n$ by defining $f(P_1, \ldots, P_n) = \{f(p_1, \ldots, p_n) \mid p_1 \in P_1, \ldots, p_n \in P_n\}$.

2 Abstract Syntax

We define the abstract syntax of a fragment of the language of UML 2.0 interactions of [2]. We assume two primitive domains for *instances* $\mathbb{I}$ and *messages* $\mathbb{M}$. An *event* e is

$$
\begin{array}{rcl}
\textit{Interaction} & ::= & \mathsf{None} \mid \mathsf{Empty} \\
 & \mid & \textit{Basic} \\
 & \mid & \textit{CombinedFragment} \\
\textit{CombinedFragment} & ::= & \mathsf{strict}(\textit{Interaction}, \textit{Interaction}) \\
 & \mid & \mathsf{seq}(\textit{Interaction}, \textit{Interaction}) \\
 & \mid & \mathsf{par}(\textit{Interaction}, \textit{Interaction}) \\
 & \mid & \mathsf{loop}(\textit{Nat}, (\textit{Nat} \mid \infty), \textit{Interaction}) \\
 & \mid & \mathsf{ignore}(\textit{Messages}, \textit{Interaction}) \\
 & \mid & \mathsf{restr}(\textit{Instances}, \textit{Interaction}) \\
 & \mid & \mathsf{alt}(\textit{Interaction}, \textit{Interaction}) \\
 & \mid & \mathsf{not}(\textit{Interaction})
\end{array}
$$

Table 1. Abstract syntax of interactions (fragment).

either of the form snd(s, r, m) or of the form rcv(s, r, m), representing the dispatch and the arrival of message m from *sender* instance s to *receiver* instance r, respectively. The set $\mathbb{E}$ comprises all events over $\mathbb{I}$ and $\mathbb{M}$. The message of an event e is denoted by $\mu(e)$. We say that the instance s is *active* for snd(s, r, m) and, similarly, that the instance r is *active* for rcv(s, r, m); the (singleton) set of instances active for an event e is denoted by $\alpha(e)$. We define a binary, symmetric *conflict* relation $\lessgtr$ on events: $e \lessgtr e' \Leftrightarrow \alpha(e) \cap \alpha(e') \neq \emptyset$.

A *basic* interaction is given by an event-labelled pomset $[(E, \leq_E, \lambda_E)]$ such that conflicting events do not occur concurrently, i.e., if $e_1, e_2 \in E$ with $\lambda_E(e_1) \lessgtr \lambda_E(e_2)$, then $e_1 \leq_E e_2$ or $e_2 \leq_E e_1$.

The abstract syntax of interactions is given by the grammar in Tab. 1. Therein, *Basic* ranges over the basic interactions, *Nat* ranges over the natural numbers, *Messages* over the subsets of $\mathbb{M}$, and *Instances* over the subsets of $\mathbb{I}$.

Note that the interaction constants None and Empty as well as the interaction operators restr and not are not part of the specification of UML 2.0 interactions as defined in [2]. The operator not results from the translation of UML 2.0 interactions (including the negative operators neg and assert) into "interactions in the classical sense"; see [1]. The operator restr and the constants None and Empty simplify the definition of the operational semantics. Interactions of the form seq(alt(B_1, B_2), B_2) motivate the operator restr, where B_i are the basic interactions $\{$snd$(s_i, r_i, m_i) \leq$ rcv$(s_i, r_i, m_i)\}$ $(i = 1, 2)$ with $m_1 \neq m_2$. In case the second operand of the weak sequencing operator seq makes progress by sending the message m_2 from instance s_2 to instance r_2, then the first operator of the seq, which is a disjunction, may only choose its first operator; we call that progress a *non-local choice*.

3 Denotational Semantics

3.1 Semantic Domains

The domain $\mathbb{P}$ comprises all basic interactions. The subdomain $\mathbb{T}$ of $\mathbb{P}$ comprises all pomsets in $\mathbb{P}$ that are traces. In particular, the empty pomset ε is in $\mathbb{T}$; also all events can be identified with traces of length one. The notion of the set of active instances of an event $e \in \mathbb{E}$ is extended to an event-labelled pomset $p = [(X, \leq_X, \lambda_X)]$ by setting $\alpha(p) = \bigcup_{x \in X} \alpha(\lambda_X(x))$.

For a pomset $p = [(X, \leq_X, \lambda_X)] \in \mathbb{P}$ and an event $e \in \mathbb{E}$ we write $e \in \min p$, if there is an $x \in X$ with $x \in \min_{\leq_X} X$ and $\lambda_X(x) = e$; note that x is unique defined, if it exists. If $e \in \min p$, we write $p \setminus \{e\}$ for $[(X \setminus \{x\}), \leq_X \cap (X \setminus \{x\})^2, \lambda_X \restriction (X \setminus \{x\})]$ with $x \in \min_{\leq_X} X$ and $\lambda_X(x) = e$.

On pomsets in $\mathbb{P}$ and for a set of messages M, the *filtering* relation $\mathit{filter}(M) : \mathbb{P} \to \wp\mathbb{P}$ removes some elements of a pomset whose labels show a message in M. More precisely, we first define $\mathit{filter}(M)$ on event-labelled sets: Let X be a set and $\lambda : X \to \mathbb{E}$ a labelling function; then $X' \in \mathit{filter}(M)(X, \lambda)$ if $X' \subseteq X$ and, if $x \in X \setminus X'$, then $\mu(\lambda(x)) \in M$. For an event-labelled partial order $(X, \leq_X, \lambda_X)$ we set $(X', \leq_X \cap (X' \times X'), \lambda_X \restriction X') \in \mathit{filter}(M)(X, \leq_X, \lambda_X)$ if $X' \in \mathit{filter}(M)(X, \lambda_X)$. Finally, we extend these definitions to event-labelled pomsets by setting $[(X', \leq_{X'}, \lambda_{X'})] \in \mathit{filter}(M)([(X, \leq_X, \lambda_X)])$ if $(X', \leq_{X'}, \lambda_{X'}) \in \mathit{filter}(M)(X, \leq_X, \lambda_X)$, which is obviously well-defined. For a pomset $p \in \mathbb{P}$, we write $p\langle M \rangle$ for $\mathit{filter}(M)^{-1}(p)$; and, consequently, for a process $P \subseteq \mathbb{P}$, we write $P\langle M \rangle$ for $\mathit{filter}(M)^{-1}(P)$.

Finally, on processes in $\wp\mathbb{P}$ and for a set of instances L, the *restriction* function $\mathit{restr}(L) : \wp\mathbb{P} \to \wp\mathbb{P}$ removes all those pomsets from a process which show an event that is active for an instance in L, i.e., $\mathit{restr}(L)(P) = \{p \in P \mid \alpha(p) \cap L = \emptyset\}$. We also write $P[L]$ for $\mathit{restr}(L)(P)$.

The process building operators are transferred to traces using the following identities:

Lemma 1. *Let $P, P_1, P_2 \subseteq \mathbb{P}$ be processes, $M \subseteq \mathbb{M}$ a set of messages, and $L \subseteq \mathbb{I}$ a set of instances.*

1. $(P_1 \mathbin{;} P_2){\downarrow} = (P_1{\downarrow}) \mathbin{;} (P_2{\downarrow})$
2. $(P_1 \mathbin{;_{\between}} P_2){\downarrow} = ((P_1{\downarrow}) \mathbin{;_{\between}} (P_2{\downarrow})){\downarrow}$
3. $(P_1 \parallel P_2){\downarrow} = ((P_1{\downarrow}) \parallel (P_2{\downarrow})){\downarrow}$
4. $(P\langle M \rangle){\downarrow} = ((P{\downarrow})\langle M \rangle){\downarrow}$
5. $(P[L]){\downarrow} = (P{\downarrow})[L]$
6. $(P_1 \cup P_2){\downarrow} = (P_1{\downarrow}) \cup (P_2{\downarrow})$
7. $(\mathbb{P} \setminus P){\downarrow} = (\mathbb{P} \setminus (P{\downarrow})){\downarrow}$

3.2 Trace-Based Semantics

The trace-based, denotational semantics of the authors [1] can be rendered as a function $\mathscr{P} : \mathit{Interaction} \to \wp\mathbb{T}$ defined as follows:

$$\mathscr{P}\mathsf{None} = \emptyset$$
$$\mathscr{P}\mathsf{Empty} = \{\varepsilon\}$$
$$\mathscr{P}B = B{\downarrow}$$
$$\mathscr{P}\mathsf{strict}(S, S') = \mathscr{P}S \mathbin{;} \mathscr{P}S'$$
$$\mathscr{P}\mathsf{seq}(S, S') = (\mathscr{P}S \mathbin{;_{\between}} \mathscr{P}S'){\downarrow}$$

$$\mathscr{P}\mathsf{par}(S, S') = (\mathscr{P}S \parallel \mathscr{P}S')\downarrow$$
$$\mathscr{P}\mathsf{loop}(m, \overline{n}, S) = \bigcup_{m \le i < \overline{n}+1} ((\mathscr{P}S)^{i_{\asymp}})\downarrow$$
$$\mathscr{P}\mathsf{ignore}(M, S) = ((\mathscr{P}S)\langle M \rangle)\downarrow$$
$$\mathscr{P}\mathsf{restr}(L, S) = (\mathscr{P}S)[L]$$
$$\mathscr{P}\mathsf{alt}(S, S') = \mathscr{P}S \cup \mathscr{P}S'$$
$$\mathscr{P}\mathsf{not}(S) = \mathbb{T} \setminus \mathscr{P}S$$

where S and S' are interactions, $M \subseteq \mathbb{M}$, $L \subseteq \mathbb{I}$, and $\overline{n}$ a natural number or infinity, where $\infty + 1 = \infty$.

In particular, a trace t is *positive* for an interaction S, written $t \models_{\mathrm{p}} S$ if, and only if, $t \in \mathscr{P}S$.

We make use of the following syntactical identifications:

$$\mathsf{strict}(\mathsf{Empty}, S) \equiv S$$
$$\mathsf{seq}(\mathsf{Empty}, S) \equiv S$$
$$\mathsf{par}(\mathsf{Empty}, S) \equiv \mathsf{par}(S, \mathsf{Empty}) \equiv S$$
$$\mathsf{loop}(0, 0, S) \equiv \mathsf{Empty}$$
$$\mathsf{restr}(L, \mathsf{Empty}) \equiv \mathsf{Empty}$$
$$\mathsf{ignore}(\emptyset, \mathsf{Empty}) \equiv \mathsf{Empty}$$
$$\mathsf{alt}(\mathsf{Empty}, \mathsf{Empty}) \equiv \mathsf{Empty}$$

4 Processes

For a process $P \subseteq \mathbb{P}$ and an event $e \in \mathbb{E}$, we define the *left quotient* $P \,/\, e$ of P by e to be the process $\{p \in \mathbb{P} \mid e\,;p \in P\}$. This operation is right-adjoined to prefixing pomsets by e with respect to set inclusion, as $e\,;P \subseteq P'$ if, and only if $P \subseteq P' \,/\, e$.

Lemma 2. *Let $P, P_1, P_2 \subseteq \mathbb{P}$ be processes, $e \in \mathbb{E}$ an event, $M \subseteq \mathbb{M}$ a set of messages, and $L \subseteq \mathbb{I}$ a set of instances.*

1. $(P_1\,; P_2) \,/\, e = ((P_1 \,/\, e)\,; P_2) \cup ((P_1 \cap \{\varepsilon\})\,; (P_2 \,/\, e))$
2. $(P_1 \,;_{\asymp} P_2) \,/\, e = ((P_1 \,/\, e) \,;_{\asymp} P_2) \cup (P_1[\alpha(e)] \,;_{\asymp} (P_2 \,/\, e))$
3. $(P_1 \parallel P_2) \,/\, e = ((P_1 \,/\, e) \parallel P_2) \cup (P_1 \parallel (P_2 \,/\, e))$
4. $(P\langle M \rangle) \,/\, e = (P \,/\, e)\langle M \rangle \cup \{\varepsilon \mid \mu(e) \in M\}\,; P\langle M \rangle$
5. $(P[L]) \,/\, e = \{\varepsilon \mid \alpha(e) \cap L = \emptyset\}\,; (P \,/\, e)[L]$
6. $(P_1 \cup P_2) \,/\, e = (P_1 \,/\, e) \cup (P_2 \,/\, e)$
7. $(\mathbb{P} \setminus P) \,/\, e = \mathbb{P} \setminus (P \,/\, e)$

Proof. By calculation, we have:

(1) $\quad (P_1\,; P_2) \,/\, e$
$$= \{p \mid \exists p_1, p_2 \,.\, p_1 \in P_1 \wedge p_2 \in P_2 \wedge e\,; p = p_1\,; p_2\}$$
$$= \{p \mid \quad \exists p_1', p_2 \,.\, e\,; p_1' \in P_1 \wedge p_2 \in P_2 \wedge p = p_1'\,; p_2$$

$$
\begin{aligned}
&\qquad \vee\, \exists p_1, p_2' \,.\, \varepsilon = p_1 \in P_1 \wedge e \,;\, p_2' \in P_2 \wedge p = p_1 \,;\, p_2'\} \\
&= (P_1 \,/\, e) \,;\, P_2 \cup (P_1 \cap \{\varepsilon\}) \,;\, (P_2 \,/\, e)
\end{aligned}
$$

(2) $(P_1 \,;_{\between}\, P_2) \,/\, e$

$$
\begin{aligned}
&= \{p \mid \exists p_1, p_2 \,.\, p_1 \in P_1 \wedge p_2 \in P_2 \wedge e \,;\, p = p_1 \,;_{\between}\, p_2\} \\
&= \{p \mid \quad \exists p_1', p_2 \,.\, e \,;\, p_1' \in P_1 \wedge p_2 \in P_2 \wedge p = p_1' \,;_{\between}\, p_2 \\
&\qquad \vee\, \exists p_1, p_2' \,.\, p_1 \in P_1 \wedge \alpha(p_1) \cap \alpha(e) = \emptyset \wedge e \,;\, p_2' \in P_2 \wedge p = p_1 \,;_{\between}\, p_2'\} \\
&= (P_1 \,/\, e) \,;_{\between}\, P_2 \cup (P_1[\alpha(e)] \,;_{\between}\, (P_2 \,/\, e))
\end{aligned}
$$

(3) $(P_1 \parallel P_2) \,/\, e$

$$
\begin{aligned}
&= \{p \mid \exists p_1, p_2 \,.\, p_1 \in P_1 \wedge p_2 \in P_2 \wedge e \,;\, p = p_1 \parallel p_2\} \\
&= \{p \mid \quad \exists p_1', p_2 \,.\, e \,;\, p_1' \in P_1 \wedge p_2 \in P_2 \wedge p = p_1' \parallel p_2 \\
&\qquad \vee\, \exists p_1, p_2' \,.\, p_1 \in P_1 \wedge e \,;\, p_2' \in P_2 \wedge p = p_1 \parallel p_2'\} \\
&= ((P_1 \,/\, e) \parallel P_2) \cup (P_1 \parallel (P_2 \,/\, e))
\end{aligned}
$$

(4) $(P\langle M\rangle) \,/\, e$

$$
\begin{aligned}
&= \{p \mid \quad \exists p' \,.\, e \,;\, p' \in P \wedge p \in p'\langle M\rangle \\
&\qquad \vee\, p \in P\langle M\rangle \wedge \mu(e) \in M\} \\
&= (P \,/\, e)\langle M\rangle \cup \{\varepsilon \mid \mu(e) \in M\} \,;\, P\langle M\rangle
\end{aligned}
$$

(5) $(P[L]) \,/\, e$

$$
\begin{aligned}
&= \{p \mid \exists p' \,.\, e \,;\, p' \in P \wedge \alpha(p') \cap \alpha(L) = \emptyset \wedge \alpha(e) \cap \alpha(L) = \emptyset \wedge p = p'\} \\
&= \{\varepsilon \mid \alpha(e) \cap L = \emptyset\} \,;\, (P \,/\, e)[L]
\end{aligned}
$$

(6) $(P_1 \cup P_2) \,/\, e$

$$
\begin{aligned}
&= \{p \mid e \,;\, p \in P_1\} \cup \{p \mid e \,;\, p \in P_2\} \\
&= (P_1 \,/\, e) \cup (P_2 \,/\, e)
\end{aligned}
$$

(7) $(\mathbb{P} \setminus P) \,/\, e$

$$
\begin{aligned}
&= \{p \mid \neg(e \,;\, p \in P)\} \\
&= \mathbb{P} \setminus (P \,/\, e)
\end{aligned}
$$

The following lemma summarises an obvious characterisation of when the empty pomset can result from a process expression:

Lemma 3. *Let $P, P_1, P_2 \subseteq \mathbb{P}$ be processes, $M \subseteq \mathbb{M}$ a set of messages, and $L \subseteq \mathbb{I}$ a set of instances.*

1. $\varepsilon \in (P_1 \,;\, P_2) \iff (\varepsilon \in P_1) \wedge (\varepsilon \in P_2)$
2. $\varepsilon \in (P_1 \,;_{\between}\, P_2) \iff (\varepsilon \in P_1) \wedge (\varepsilon \in P_2)$
3. $\varepsilon \in (P_1 \parallel P_2) \iff (\varepsilon \in P_1) \wedge (\varepsilon \in P_2)$
4. $\varepsilon \in (P\langle M\rangle) \iff \varepsilon \in P$
5. $\varepsilon \in (P[L]) \iff \varepsilon \in P$
6. $\varepsilon \in (P_1 \cup P_2) \iff (\varepsilon \in P_1) \vee (\varepsilon \in P_2)$
7. $\varepsilon \in (\mathbb{P} \setminus P) \iff \varepsilon \notin P$

Process expressions satisfy some obvious monotonicity conditions:

Lemma 4. *Let $P, P', P_1, P_1', P_2, P_2' \subseteq \mathbb{P}$ be processes, $M, M' \subseteq \mathbb{M}$ a set of messages, and $L, L' \subseteq \mathbb{I}$ sets of instances.*

1. $P_1 \subseteq P_1' \land P_2 \subseteq P_2' \Rightarrow (P_1 \,;\, P_2) \subseteq (P_1' \,;\, P_2')$
2. $P_1 \subseteq P_1' \land P_2 \subseteq P_2' \Rightarrow (P_1 \,;_{\asymp}\, P_2) \subseteq (P_1' \,;_{\asymp}\, P_2')$
3. $P_1 \subseteq P_1' \land P_2 \subseteq P_2' \Rightarrow (P_1 \parallel P_2) \subseteq (P_1' \parallel P_2')$
4. $P \subseteq P' \land M \subseteq M' \Rightarrow (P\langle M\rangle) \subseteq (P'\langle M'\rangle)$
5. $P \subseteq P' \land L \supseteq L' \Rightarrow (P[L]) \subseteq (P'[L'])$
6. $P_1 \subseteq P_1' \land P_2 \subseteq P_2' \Rightarrow (P_1 \cup P_2) \subseteq (P_1' \cup P_2')$
7. $P \supseteq P' \Rightarrow (\mathbb{P} \setminus P) \subseteq (\mathbb{P} \setminus P')$

All these observations can be extended straightforwardly to $P^{n_{\asymp}} \,/\, e$:

$$P^{0_{\asymp}} \,/\, e = \emptyset$$
$$P^{(n+1)_{\asymp}} \,/\, e = ((P \,/\, e) \,;_{\asymp}\, P^{n_{\asymp}}) \cup (P[\alpha(e)] \,;_{\asymp}\, (P^{n_{\asymp}} \,/\, e))$$
$$\varepsilon \in P^{n_{\asymp}} \iff (\varepsilon \in P) \lor (n = 0)$$
$$P \subseteq P' \Rightarrow P^{n_{\asymp}} \subseteq P'^{n_{\asymp}}$$

5 Operational Semantics

We define the domain $\mathbb{E}_\tau$ of events and the *silent event* τ as $\mathbb{E} \cup \{\tau\}$. Analogously, $\mathbb{P}_\tau$ is the domain of all pomsets labelled with events from $\mathbb{E}_\tau$, and $\mathbb{T}_\tau$ the subdomain comprising all pomsets in $\mathbb{P}_\tau$ that are traces. We define the set of active instances of τ as $\alpha(\tau) = \emptyset$.

Based on the observations in Lemma 3, we define a predicate $\varepsilon(-)$ on interactions which determines whether an interaction contains the empty trace:

$$\varepsilon(\mathsf{None}) \iff \mathit{ff}$$
$$\varepsilon(\mathsf{Empty}) \iff \mathit{tt}$$
$$\varepsilon(B) \iff B = \varepsilon$$
$$\varepsilon(\mathsf{strict}(S, S')) \iff \varepsilon(S) \land \varepsilon(S')$$
$$\varepsilon(\mathsf{seq}(S, S')) \iff \varepsilon(S) \land \varepsilon(S')$$
$$\varepsilon(\mathsf{par}(S, S')) \iff \varepsilon(S) \land \varepsilon(S')$$
$$\varepsilon(\mathsf{loop}(m, \overline{n}, S)) \iff \varepsilon(S) \lor (m = 0)$$
$$\varepsilon(\mathsf{ignore}(M, S)) \iff \varepsilon(S)$$
$$\varepsilon(\mathsf{restr}(L, S)) \iff \varepsilon(S)$$
$$\varepsilon(\mathsf{alt}(S, S')) \iff \varepsilon(S) \lor \varepsilon(S')$$
$$\varepsilon(\mathsf{not}(S)) \iff \neg\varepsilon(S)$$

The operational semantics of interactions is given by two ternary relations between interactions S and S' and an event $\overline{e} \in \mathbb{E}_\tau$: The *positive* reduction relation, denoted by

$S \xrightarrow{\overline{e}}_{\mathrm{p}} S'$, is defined by the rules in Tab. 2. The *negative* reduction relation, denoted by $S \xrightarrow{\overline{e}}_{\mathrm{n}} S'$, is defined by the rules in Tab. 3. In these rules, the variously decorated meta-variables range as follows: S over interactions, B over basic interactions, $\overline{e}$ over $\mathbb{E}_\tau$, e over $\mathbb{E}$, m over the natural numbers, $\overline{n}$ over the natural numbers or infinity.

$$(\text{basic}_{\mathrm{p}})\quad B \xrightarrow{e}_{\mathrm{p}} B \setminus \{e\} \quad \text{if } e \in \min B$$

$$(\text{strict}_{\mathrm{p}})\quad \frac{S_1 \xrightarrow{\overline{e}}_{\mathrm{p}} S_1'}{\mathsf{strict}(S_1, S_2) \xrightarrow{\overline{e}}_{\mathrm{p}} \mathsf{strict}(S_1', S_2)}$$

$$(\text{seq}^1_{\mathrm{p}})\quad \frac{S_1 \xrightarrow{\overline{e}}_{\mathrm{p}} S_1'}{\mathsf{seq}(S_1, S_2) \xrightarrow{\overline{e}}_{\mathrm{p}} \mathsf{seq}(S_1', S_2)}$$

$$(\text{seq}^2_{\mathrm{p}})\quad \frac{S_2 \xrightarrow{\overline{e}}_{\mathrm{p}} S_2'}{\mathsf{seq}(S_1, S_2) \xrightarrow{\overline{e}}_{\mathrm{p}} \mathsf{seq}(\mathsf{restr}(\alpha(\overline{e}), S_1), S_2')}$$

$$(\text{par}^1_{\mathrm{p}})\quad \frac{S_1 \xrightarrow{\overline{e}}_{\mathrm{p}} S_1'}{\mathsf{par}(S_1, S_2) \xrightarrow{\overline{e}}_{\mathrm{p}} \mathsf{par}(S_1', S_2)} \qquad (\text{par}^2_{\mathrm{p}})\quad \frac{S_2 \xrightarrow{\overline{e}}_{\mathrm{p}} S_2'}{\mathsf{par}(S_1, S_2) \xrightarrow{\overline{e}}_{\mathrm{p}} \mathsf{par}(S_1, S_2')}$$

$$(\text{loop}^1_{\mathrm{p}})\quad \mathsf{loop}(0, \overline{n}, S) \xrightarrow{\tau}_{\mathrm{p}} \mathsf{Empty}$$

$$(\text{loop}^2_{\mathrm{p}})\quad \frac{S \xrightarrow{\overline{e}}_{\mathrm{p}} S'}{\mathsf{loop}(m, \overline{n}+1, S) \xrightarrow{\overline{e}}_{\mathrm{p}} \mathsf{seq}(S', \mathsf{loop}(m \dot{-} 1, \overline{n}, S))}$$

$$(\text{ignore}^1_{\mathrm{p}})\quad \mathsf{ignore}(M, \mathsf{Empty}) \xrightarrow{\tau}_{\mathrm{p}} \mathsf{Empty} \qquad (\text{ignore}^2_{\mathrm{p}})\quad \frac{S \xrightarrow{\overline{e}}_{\mathrm{p}} S'}{\mathsf{ignore}(M, S) \xrightarrow{\overline{e}}_{\mathrm{p}} \mathsf{ignore}(M, S')}$$

$$(\text{ignore}^3_{\mathrm{p}})\quad \mathsf{ignore}(M, S) \xrightarrow{e}_{\mathrm{p}} \mathsf{ignore}(M, S) \quad \text{if } \mu(e) \in M$$

$$(\text{restr}_{\mathrm{p}})\quad \frac{S \xrightarrow{\overline{e}}_{\mathrm{p}} S'}{\mathsf{restr}(L, S) \xrightarrow{\overline{e}}_{\mathrm{p}} \mathsf{restr}(L, S')} \quad \text{if } \alpha(\overline{e}) \cap L = \emptyset$$

$$(\text{alt}^1_{\mathrm{p}})\quad \frac{S_1 \xrightarrow{\overline{e}}_{\mathrm{p}} S_1'}{\mathsf{alt}(S_1, S_2) \xrightarrow{\overline{e}}_{\mathrm{p}} S_1'} \qquad (\text{alt}^2_{\mathrm{p}})\quad \frac{S_2 \xrightarrow{\overline{e}}_{\mathrm{p}} S_2'}{\mathsf{alt}(S_1, S_2) \xrightarrow{\overline{e}}_{\mathrm{p}} S_2'}$$

$$(\text{not}^1_{\mathrm{p}})\quad \frac{S \xrightarrow{\overline{e}}_{\mathrm{n}} S'}{\mathsf{not}(S) \xrightarrow{\overline{e}}_{\mathrm{p}} \mathsf{not}(S')} \qquad (\text{not}^2_{\mathrm{p}})\quad \mathsf{not}(S) \xrightarrow{\tau}_{\mathrm{p}} \mathsf{Empty} \quad \text{if } \neg\varepsilon(S)$$

Table 2. Positive reduction relation of the operational semantics.

6 Correctness

If $S \xrightarrow{\overline{e}_1}_{\mathrm{p}} S_1$, $S_1 \xrightarrow{\overline{e}_2}_{\mathrm{p}} S_2$, $\dots$, $S_{n-1} \xrightarrow{\overline{e}_n}_{\mathrm{p}} S'$, we write $S \xrightarrow{\overline{t}}_{\mathrm{p}} S'$, where $\overline{t} = \overline{e}_1 ; \overline{e}_2 ; \cdots ; \overline{e}_n \in \mathbb{T}_\tau$ is a finite trace possibly containing one or more occurrences of

$$(\mathrm{empty_n})\ \textsf{Empty} \xrightarrow{e}_{\mathrm{n}} \textsf{None} \qquad (\mathrm{none_n})\ \textsf{None} \xrightarrow{e}_{\mathrm{n}} \textsf{None}$$

$$(\mathrm{basic}^1_{\mathrm{n}})\ B \xrightarrow{e}_{\mathrm{n}} B \setminus \{e\} \quad \text{if } e \in \min B \qquad (\mathrm{basic}^2_{\mathrm{n}})\ B \xrightarrow{e}_{\mathrm{n}} \textsf{None} \quad \text{if } e \notin \min B$$

$$(\mathrm{strict_n})\ \frac{S_1 \xrightarrow{\overline{e}}_{\mathrm{n}} S_1' \quad S_2 \xrightarrow{\overline{e}}_{\mathrm{n}} S_2'}{\textsf{strict}(S_1, S_2) \xrightarrow{\overline{e}}_{\mathrm{n}} \textsf{alt}(\textsf{strict}(S_1', S_2), \textsf{strict}(\textsf{restr}(\mathbb{I}, S_1), S_2'))}$$

$$(\mathrm{seq_n})\ \frac{S_1 \xrightarrow{\overline{e}}_{\mathrm{n}} S_1' \quad S_2 \xrightarrow{\overline{e}}_{\mathrm{n}} S_2'}{\textsf{seq}(S_1, S_2) \xrightarrow{\overline{e}}_{\mathrm{n}} \textsf{alt}(\textsf{seq}(S_1', S_2), \textsf{seq}(\textsf{restr}(\alpha(e), S_1), S_2'))}$$

$$(\mathrm{par_n})\ \frac{S_1 \xrightarrow{\overline{e}}_{\mathrm{n}} S_1' \quad S_2 \xrightarrow{\overline{e}}_{\mathrm{n}} S_2'}{\textsf{par}(S_1, S_2) \xrightarrow{\overline{e}}_{\mathrm{n}} \textsf{alt}(\textsf{par}(S_1', S_2), \textsf{par}(S_1, S_2'))}$$

$$(\mathrm{loop}^1_{\mathrm{n}})\ \textsf{loop}(0, \infty, S) \xrightarrow{\overline{e}}_{\mathrm{n}} \textsf{not}(\textsf{None})$$

$$(\mathrm{loop}^2_{\mathrm{n}})\ \frac{S \xrightarrow{\overline{e}}_{\mathrm{n}} S' \quad \textsf{loop}(m \dot{-} 1, \overline{n}, S) \xrightarrow{\overline{e}}_{\mathrm{n}} S''}{\textsf{loop}(m, \overline{n}+1, S) \xrightarrow{\overline{e}}_{\mathrm{n}} \textsf{alt}(\textsf{seq}(S', \textsf{loop}(m \dot{-} 1, \overline{n}, S)), \textsf{seq}(\textsf{restr}(\alpha(e), S), S''))}$$

$$(\mathrm{ignore}^1_{\mathrm{n}})\ \frac{S \xrightarrow{e}_{\mathrm{n}} S'}{\textsf{ignore}(M, S) \xrightarrow{e}_{\mathrm{n}} \textsf{alt}(\textsf{ignore}(M, S'), \textsf{ignore}(M, S))} \quad \text{if } \mu(e) \in M$$

$$(\mathrm{ignore}^2_{\mathrm{n}})\ \frac{S \xrightarrow{\overline{e}}_{\mathrm{n}} S'}{\textsf{ignore}(M, S) \xrightarrow{\overline{e}}_{\mathrm{n}} \textsf{ignore}(M, S')} \quad \text{if } \overline{e} = \tau \vee \mu(\overline{e}) \notin M$$

$$(\mathrm{restr}^1_{\mathrm{n}})\ \frac{S \xrightarrow{\overline{e}}_{\mathrm{n}} S'}{\textsf{restr}(L, S) \xrightarrow{\overline{e}}_{\mathrm{n}} \textsf{restr}(L, S')} \quad \text{if } \alpha(e) \cap L = \emptyset$$

$$(\mathrm{restr}^2_{\mathrm{n}})\ \textsf{restr}(L, S) \xrightarrow{e}_{\mathrm{n}} \textsf{None} \quad \text{if } \alpha(e) \cap L \neq \emptyset$$

$$(\mathrm{alt_n})\ \frac{S_1 \xrightarrow{\overline{e}}_{\mathrm{n}} S_1' \quad S_2 \xrightarrow{\overline{e}}_{\mathrm{n}} S_2'}{\textsf{alt}(S_1, S_2) \xrightarrow{\overline{e}}_{\mathrm{n}} \textsf{alt}(S_1', S_2')}$$

$$(\mathrm{not_n})\ \frac{S \xrightarrow{\overline{e}}_{\mathrm{p}} S'}{\textsf{not}(S) \xrightarrow{\overline{e}}_{\mathrm{n}} \textsf{not}(S')}$$

Table 3. Negative reduction relation of the operational semantics.

the silent event τ. Given a trace $\overline{t} \in \mathbb{T}_\tau$, we let $\lfloor \overline{t} \rfloor$ denote the trace obtained from $\overline{t}$ by removing every occurrence of the silent event τ.

The above introduced operational semantics of interactions is correct w.r.t. the denotational one, that is, given an interaction, traces that lead the interaction to Empty are positive for the interaction:

Lemma 5. *Let S, S' be interactions and $e \in \mathbb{E}$.*

1. *If $S \xrightarrow{e}_{\mathrm{p}} S'$, then $\mathscr{P} S' \subseteq \mathscr{P} S \,/\, e$.*
2. *If $S \xrightarrow{\tau}_{\mathrm{p}} S'$, then $\mathscr{P} S' \subseteq \mathscr{P} S$.*
3. *If $S \xrightarrow{e}_{\mathrm{n}} S'$, then $\mathscr{P} S' \supseteq \mathscr{P} S \,/\, e$.*

4. *If* $S \xrightarrow{\tau}_{\mathrm{n}} S'$*, then* $\mathscr{P}S' \supseteq \mathscr{P}S$.

Proof. Claims (1) and (3) follow immediately from Lemma 2, claims (2) and (4) from Lemma 4.

Proposition 6. *Let* S *be an interaction and* $\bar{t}$ *be a trace in* $\mathbb{T}_\tau$*. If* $S \xrightarrow{\bar{t}}_{\mathrm{p}} \mathsf{Empty}$*, then* $\lfloor\bar{t}\rfloor \models_{\mathrm{p}} S$.

Proof. From Lemma 5, and by induction on the length of $\bar{t}$, follows that if $S \xrightarrow{\bar{t}}_{\mathrm{p}} \mathsf{Empty}$, then $\lfloor\bar{t}\rfloor \,;\, \mathscr{P}\mathsf{Empty} \subseteq \mathscr{P}S$, i.e., $\lfloor\bar{t}\rfloor \in \mathscr{P}(S)$.

7 Completeness

Lemma 7. *Let* S *be an interaction and* $e \in \mathbb{E}$*. Then*

1. $(\mathscr{P}S) \,/\, e = \bigcup\{\mathscr{P}(S') \mid S \xrightarrow{\tau^*;e}_{\mathrm{p}} S'\}$
2. $(\mathbb{T} \setminus \mathscr{P}S) \,/\, e = \bigcap\{\mathbb{T} \setminus \mathscr{P}(S') \mid S \xrightarrow{\tau^*;e}_{\mathrm{n}} S'\}$
3. $\varepsilon \in \mathscr{P}S \iff S \xrightarrow{\tau^*}_{\mathrm{p}} \mathsf{Empty}$

Proof. Claims (1) and (2) follow by mutual induction on the term structure of the interaction S from Lemma 2.

Claim (3) follows by induction on the term structure of the interaction S from the syntactical identifications, the definition of $\varepsilon(-)$ and the τ-rules.

Proposition 8. *Let* S *be an interaction and* $t \in \mathbb{T}$ *be a finite trace. If* $t \in \mathscr{P}(S)$*, then there is a* $\bar{t} \in \mathbb{T}_\tau$ *with* $t = \lfloor\bar{t}\rfloor$ *and* $S \xrightarrow{\bar{t}}_{\mathrm{p}} \mathsf{Empty}$.

Proof. From Lemma 7(1) and by induction follows that for every finite trace $t \in \mathbb{T}$ and for every interaction S with $t \in \mathscr{P}(S)$, there are a $\bar{t} \in \mathbb{T}_\tau$ with $t = \lfloor\bar{t}\rfloor$ and an interaction S_t such that $S \xrightarrow{\bar{t}}_{\mathrm{p}} S_t$ and $\varepsilon \in \mathscr{P}S_t$. Thus, by Lemma 7(3), for an interaction S and a finite trace $t \in \mathbb{T}$, there is a $\bar{t} \in \mathbb{T}_\tau$ with $t = \lfloor\bar{t}\rfloor$ and $S \xrightarrow{\bar{t}}_{\mathrm{p}} \mathsf{Empty}$.

References

1. María Victoria Cengarle and Alexander Knapp. UML 2.0 Interactions: Semantics and Refinement. In Jan Jürjens, Eduardo B. Fernandez, Robert France, and Bernhard Rumpe, editors, *Proc. 3rd Int. Wsh. Critical Systems Development with UML (CSDUML'04)*, pages 85–99. Technical Report TUM-I0415, Institut für Informatik, Technische Universität München, 2004.
2. Object Management Group. UML 2.0 Superstructure Specification. Final adopted specification, OMG, 2003. `http://www.omg.org/cgi-bin/doc?ptc/03-08-02`.
3. Vaughan Pratt. Modeling Concurrency with Partial Orders. *Int. J. Parallel Program.*, 15(1):33–71, 1986.