

RoboScene to CSP Translation Rules

Holly Hendry

June 13, 2024

$$\begin{aligned} & \llbracket \text{caps} : \text{Capabilities}; \text{seqdgrmgrps} : \text{SequenceDiagramGroup} \rrbracket_D \\ & \hat{=} \llbracket \text{caps} \rrbracket_{CS} \llbracket \text{seqdgrmgrps} \rrbracket_{SDS} \end{aligned}$$

$$\begin{aligned} & \llbracket \text{def} \rrbracket_{SDS} \hat{=} \llbracket \text{def} \rrbracket_{SD} \\ & \llbracket \text{def defS} \rrbracket_{SDS} \hat{=} \llbracket \text{def} \rrbracket_{SD} \llbracket \text{defS} \rrbracket_{SDS} \end{aligned}$$

$$\begin{aligned} & \llbracket \text{constant} : \text{Constants}; \text{lifelines} : \text{LifelineBlock} \rrbracket_{CS} \hat{=} \\ & \llbracket \text{constant} \rrbracket_{CO} \llbracket \text{lifelines} \rrbracket_{LS} \end{aligned}$$

$$\begin{aligned} & \llbracket \text{timeunit} : \text{TimeUnit}, \text{consts} : \text{VariableList} \rrbracket_{CO} \\ & \hat{=} \llbracket \text{timeunit} \rrbracket_{TU} \llbracket \text{consts} \rrbracket_{CONS} \end{aligned}$$

$$\begin{aligned} & \llbracket \text{def} \rrbracket_{CONS} \hat{=} \llbracket \text{def} \rrbracket_{CON} \\ & \llbracket \text{def defS} \rrbracket_{CONS} \hat{=} \llbracket \text{def} \rrbracket_{CON} \llbracket \text{defS} \rrbracket_{CONS} \end{aligned}$$

$$\llbracket (\text{const } \text{constID} = \text{val}) : \text{VariableList} \rrbracket_{CON} \hat{=} \text{constID} = \text{val}$$

$$\begin{aligned}
& \llbracket \text{lifeline} : \text{Actor}; \text{ins}, \text{outs} : \text{LifelineEvents}; \text{vars} : \text{VarsList} \rrbracket_{LS} \\
& \quad \hat{=} \llbracket \text{outs} \rrbracket_{OS} \llbracket \text{ins} \rrbracket_{IS} \llbracket \text{vars} \rrbracket_{VS} \\
\\
& \llbracket \text{def} \rrbracket_{OS} = \llbracket \text{def} \rrbracket_O \\
& \llbracket \text{def defS} \rrbracket_{OS} \hat{=} \llbracket \text{def} \rrbracket_O \llbracket \text{defS} \rrbracket_{OS} \\
& \llbracket \text{def} \rrbracket_{IS} \hat{=} \llbracket \text{def} \rrbracket_I \\
& \llbracket \text{def defS} \rrbracket_{IS} \hat{=} \llbracket \text{def} \rrbracket_I \llbracket \text{defS} \rrbracket_{IS} \\
& \llbracket \text{def} \rrbracket_{VS} \hat{=} \llbracket \text{def} \rrbracket_V \\
& \llbracket \text{def}, \text{defS} \rrbracket_{VS} \hat{=} \llbracket \text{def} \rrbracket_V \llbracket \text{defS} \rrbracket_{VS} \\
\\
& \llbracket (\text{out event} \rightarrow \text{actor}) : \text{LifelineEvents} \rrbracket_O \hat{=} \textit{channel event} : \underline{\text{InOut}} \\
& \llbracket (\text{out event} : \text{Type} \rightarrow \text{actor}) : \text{LifelineEvents} \rrbracket_O \\
& \quad \hat{=} \textit{channel event} : \llbracket \text{Type}, \text{event} \rrbracket_{TO} \\
& \llbracket (\text{in event} < \text{actor}) : \text{LifelineEvents} \rrbracket_I \hat{=} \textit{channel event} : \underline{\text{InOut}} \\
& \llbracket (\text{in event} : \text{Type} < \text{actor}) : \text{LifelineEvents} \rrbracket_I \\
& \quad \hat{=} \textit{channel event} : \llbracket \text{Type}, \text{event} \rrbracket_{TI} \\
\\
& \llbracket (\text{var} : \text{Type}) : \text{VariableList} \rrbracket_V \hat{=} \\
& \quad \textit{channel set_var} : \text{Type} \\
& \quad \textit{channel get_var} : \text{Type} \\
& \quad \textit{Memory_var}(\text{var}) = \textit{get_var}! \text{var} \rightarrow \textit{Memory_var}(\text{var}) \\
& \quad \quad \square \textit{set_var}?x \rightarrow \textit{Memory_var}(x) \\
& \quad \quad \square \textit{terminate} \rightarrow \textit{Skip} \\
& \quad \textit{varMemory} = \underline{(\parallel \text{var} \bullet \{\textit{terminate}\} \circ \textit{Memory_var}(\text{var}))}
\end{aligned}$$

$$\llbracket \text{name : EString; trait : EString; actors : Actor; sequences : Sequence} \rrbracket_{SD} \\ \hat{=} \llbracket \text{actors} \rrbracket_A \llbracket \text{sequences} \rrbracket_{SS}$$

$$\llbracket \text{def} \rrbracket_{SS} \hat{=} \llbracket \text{def} \rrbracket_S \\ \llbracket \text{def defS} \rrbracket_{SS} \hat{=} \llbracket \text{def} \rrbracket_S \llbracket \text{defS} \rrbracket_{SS}$$

$$\llbracket \text{actors : Actor; vars : VarsList; frags : InteractionFragment} \rrbracket_S \hat{=} \llbracket \text{frags} \rrbracket_{FS} \\ (((\llbracket \text{a : actors(a}_1, \dots, \text{a}_n) \bullet \text{alpha(a)} \circ \text{lifeline(a)} \rrbracket \\ \llbracket \{ \text{str, par, terminate} \} \rrbracket \text{control}(\text{parFragS, strFragS}) \\ \llbracket \{ \text{alt, opt, loop, guard, terminate} \} \rrbracket \text{guards}(\text{altFragS, loopFragS, optFragS}) \\ \llbracket \text{sharedVars} \cup \{ \text{terminate} \} \rrbracket \text{varMemory}) \setminus \{ \text{alt, opt, loop, guard, str, par} \}$$

$$\underline{\text{alpha(a)}} = \text{diff}((\alpha(a_x, \text{frags}) \cap \alpha(a_{x+1}, \text{frags})), \underline{\text{gets}}())$$

$$\underline{\text{lifeline(a)}} = \llbracket \text{frags}_1 \rrbracket_{(a)}^F \circ \dots \circ \llbracket \text{frags}_n \rrbracket_{(a)}^F$$

$$\underline{\text{control}(\text{parFragS, strFragS})} = \underline{\text{parallel}(\text{parFragS})} \llbracket \{ \text{terminate} \} \rrbracket \underline{\text{strict}(\text{strFragS})}$$

$$\textbf{channel alt} : \text{IDs.Int}$$

$$\textbf{channel opt} : \text{IDs.Int}$$

$$\textbf{channel loop} : \text{IDs.Int}$$

$$\textbf{channel par} : \text{IDs.Int}$$

$$\textbf{channel str} : \text{IDs.Int}$$

$$\textbf{channel guard} : \text{IDs.Int.Int.Bool}$$

$$\text{datatype IDs} = \text{ID_ALT} \mid \text{ID_OPT} \mid \text{ID_LOOP} \mid \text{ID_PAR} \mid \text{ID_STR}$$

$$\llbracket \text{frags} \rrbracket_{FS} \hat{=} \llbracket \text{frag} \rrbracket_F$$

$$\llbracket \text{frag frags} \rrbracket_{FS} \hat{=} \llbracket \text{frag} \rrbracket_F \llbracket \text{frags} \rrbracket_{FS}$$

$$\llbracket (\text{wait(x)} \text{ on } \text{a}) : \text{WaitOccurrence} \rrbracket_F = \text{Wait(x)}$$

$$\llbracket (\text{wait}(\llbracket \text{v}, \text{y} \rrbracket) \text{ on } \text{a}) : \text{WaitOccurrence} \rrbracket_F = (\sqcap x : \{ \text{v..y} \} \bullet \text{wait}(x))$$

$$\llbracket (\text{deadline(x)} \text{ b}) : \text{DeadlineFragment} \rrbracket_F = \text{Deadline(b, x)}$$

$$\llbracket (\text{destroy on a}) : \text{DestroyOccurrence} \rrbracket_F = \text{terminate} \rightarrow \textbf{Skip}$$

$$\begin{aligned} & \text{guards}(\text{altFrag}, \text{loopFrag}, \text{optFrag}) = \\ & \quad (\text{evaluation}(\text{altFrag}, \text{loopFrag}, \text{optFrag}) \parallel [\text{alphaCounters} \cup \{\text{terminate}\}] \parallel \text{counters}) \\ & \quad \setminus \text{alphaCounters} \end{aligned}$$

channel $\text{getCount} : ID.Int.Int$
channel $\text{setCount} : ID.Int.Int$
 $\text{alphaCounters} = \{\text{getCount}, \text{setCount}\}$

$$\begin{aligned} & \text{Counter_ID_x}(\text{count}) = \text{getCount.ID.x!count} \rightarrow \text{Counter_ID_x}(\text{count}) \\ & \quad \square \text{setCount.ID.x?y} \rightarrow \text{Counter_ID_x}(y) \\ & \quad \square \text{terminate} \rightarrow \mathbf{Skip} \\ & \text{counters} = (\parallel ID : IDs, x : \{0..n\} \bullet \{\text{terminate}\} \circ \text{Counters_ID_x}(0)) \end{aligned}$$

$$\begin{aligned} & \text{evaluation}(\text{altFrag}, \text{loopFrag}, \text{optFrag}) = \\ & \quad \text{reset_counters} \circ \text{guards_response}(\text{altFrag}, \text{loopFrag}, \text{optFrag}) \\ & \text{reset_counters} = \circ ID : IDs, x : \{0..n\} \bullet \text{setCount.ID.x!0} \rightarrow \mathbf{Skip} \end{aligned}$$

$$\begin{aligned} & \text{guards_response}(\text{altFrag}, \text{loopFrag}, \text{optFrag}) = \\ & \quad \square \text{alt?ID_ALT.id} \rightarrow (\frac{[[\text{altFrag} : \text{XAltFragment}]]_{AF} : \text{altFrag} \bullet}{(\text{id} = \text{id}(\text{altFrag})) \& (\text{alt_eval}([[\text{altFrag}]]_{AF}))} \circ \text{guards_response}) \\ & \quad \square \text{opt?ID_OPT.id} \rightarrow (\frac{[[\text{optFrag} : \text{OptFragment}]]_{OF} : \text{optFrag} \bullet}{(\text{id} = \text{id}(\text{optFrag})) \& (\text{opt_eval}([[\text{optFrag}]]_{OF}))} \circ \text{guards_response}) \\ & \quad \square \text{loop?ID_LOOP.id} \rightarrow (\frac{[[\text{loopFrag} : \text{LoopFragment}]]_{LF} : \text{loopFrag} \bullet}{(\text{id} = \text{id}(\text{loopFrag})) \& (\text{loop_eval}([[\text{loopFrag}]]_{LF}))} \circ \text{guards_response}) \\ & \quad \square \text{terminate} \rightarrow \mathbf{Skip} \end{aligned}$$

$$\begin{aligned} & \text{alt_eval}(\text{altFrag}) \hat{=} \\ & \quad \frac{(\text{variable} : \text{getFragmentGuardVariables}([[\text{altFrag}]]_{AF}) \bullet}{(\text{get_}\{\text{variable}\} \rightarrow)) ([[\text{altFrag}]]_G)} \end{aligned}$$

$$\text{alt_}(a, x) = [[\text{altFrag}]]_{AF}$$

$$\begin{aligned} & \text{opt_eval}(\text{optFrag}) \hat{=} \\ & \quad \frac{(\text{variable} : \text{getFragmentGuardVariables}([[\text{optFrag}]]_{OF}) \bullet}{(\text{get_}\{\text{variable}\} \rightarrow)) ([[\text{optFrag}]]_G)} \end{aligned}$$

$$\text{opt_}(a, x) = [[\text{optFrag}]]_{OF}$$

$$\begin{aligned} & \text{loop_eval}(\text{loopFrag}) \hat{=} \\ & \quad \frac{(\text{variable} : \text{getFragmentGuardVariables}([[\text{loopFrag}]]_{LF}) \bullet}{(\text{get_}\{\text{variable}\} \rightarrow)) ([[\text{loopFrag}]]_G)} \end{aligned}$$

$$\text{loop_}(a, x) = [[\text{loopFrag}]]_{LF}$$

$$\begin{aligned}
\llbracket \text{altFrag} : \text{XAltFragment} \rrbracket_F &= \text{alt.ID_ALT}.\underline{\text{id}(\text{altFrag})} \rightarrow \textit{Skip} \\
\llbracket \text{optFrag} : \text{OptFragment} \rrbracket_F &= \text{opt.ID_OPT}.\underline{\text{id}(\text{optFrag})} \rightarrow \textit{Skip} \\
\llbracket \text{loopFrag} : \text{LoopFragment} \rrbracket_F &= \text{loop.ID_LOOP}.\underline{\text{id}(\text{loopFrag})} \rightarrow \textit{Skip} \\
\llbracket \text{strFrag} : \text{StrFragment} \rrbracket_F &= \text{str.ID_STR}.\underline{\text{id}(\text{strFrag})} \rightarrow \textit{Skip} \\
\llbracket \text{parFrag} : \text{ParFragment} \rrbracket_F &= \text{par.ID_PAR}.\underline{\text{id}(\text{parFrag})} \rightarrow \textit{Skip}
\end{aligned}$$

$$\begin{aligned}
\llbracket \text{loopFrag} : \text{LoopFragment} \rrbracket_G = & \\
& \text{not}(\text{condition}) \& \\
& (\text{setCount.ID_LOOP.id}(\text{loopFrag})!0 \rightarrow \\
& \text{guard.ID_LOOP.id}(\text{loopFrag}).1!\text{false} \rightarrow \text{Skip}) \\
& \square \\
& ((\text{count} < \text{min}) \text{and } \text{condition}) \& \\
& (\text{setCount.ID_LOOP.id}(\text{loopFrag})!(\text{count} + 1) \rightarrow \\
& \text{guard.ID_LOOP.id}(\text{loopFrag}).1!\text{true} \rightarrow \text{Skip}) \\
& \square \\
& (\text{count} == \text{max}) \& \\
& (\text{setCount.ID_LOOP.id}(\text{loopFrag})!0 \rightarrow \\
& \text{guard.ID_LOOP.id}(\text{loopFrag}).1!\text{false} \rightarrow \text{Skip}) \\
& \square \\
& (\text{not}(\text{count} < \text{min}) \text{ and } \text{not}(\text{count} \geq \text{max}) \text{and } \text{condition}) \& \\
& (\text{setCount.ID_LOOP.id}(\text{loopFrag})!(\text{count} + 1) \rightarrow \\
& \text{guard.ID_LOOP.id}(\text{loopFrag}).1!\text{true} \rightarrow \text{Skip}) \\
& \square \\
& ((\text{count} \geq \text{min}) \text{ and } (\text{count} < \text{max}) \text{ and } \text{condition}) \& \\
& ((\text{setCount.ID_LOOP.id}(\text{loopFrag})!(\text{count} + 1) \rightarrow \\
& \text{guard.ID_LOOP.id}(\text{loopFrag}).1!\text{true} \rightarrow \text{Skip}) \sqcap \\
& (\text{setCount.ID_LOOP.id}(\text{loopFrag})!0 \rightarrow \\
& \text{guard.ID_LOOP.id}(\text{loopFrag}).1!\text{false} \rightarrow \text{Skip}))), \\
\text{if } \text{loopFrag} \in \llbracket (\text{loop}(\text{min}, \text{max})[\text{condition}]) : \text{LoopFragment} \rrbracket_{LF}
\end{aligned}$$

$$\begin{aligned}
\llbracket \text{loopFrag} : \text{LoopFragment} \rrbracket_G = & \\
& \text{getCount.ID_LOOP.id}(\text{loopFrag})?\text{count} \rightarrow (\\
& (\text{count} < \text{min}) \& \\
& (\text{setCount.ID_LOOP.id}(\text{loopFrag})!(\text{count} + 1) \rightarrow \\
& \text{guard.ID_LOOP.id}(\text{loopFrag}).1!\text{true} \rightarrow \text{Skip}) \\
& \square \\
& (\text{count} == \text{defaultMax}) \& \\
& (\text{setCount.ID_LOOP.id}(\text{loopFrag})!0 \rightarrow \\
& \text{guard.ID_LOOP.id}(\text{loopFrag}).1!\text{false} \rightarrow \text{Skip}) \\
& \square \\
& (\text{not}(\text{count} < \text{min}) \text{ and } \text{not}(\text{count} \geq \text{defaultMax})) \& \\
& (\text{setCount.ID_LOOP.id}(\text{loopFrag})!(\text{count} + 1) \rightarrow \\
& \text{guard.ID_LOOP.id}(\text{loopFrag}).1!\text{true} \rightarrow \text{Skip}) \\
& \square \\
& (\text{count} \geq \text{min} \text{ and } \text{count} < \text{defaultMax}) \& \\
& ((\text{setCount.ID_LOOP.id}(\text{loopFrag})!(\text{count} + 1) \rightarrow \\
& \text{guard.ID_LOOP.id}(\text{loopFrag}).1!\text{true} \rightarrow \text{Skip}) \sqcap \\
& (\text{setCount.ID_LOOP.id}(\text{loopFrag})!0 \rightarrow \\
& \text{guard.ID_LOOP.id}(\text{loopFrag}).1!\text{false} \rightarrow \text{Skip}))), \\
\text{if } \text{loopFrag} \in \llbracket (\text{loop}(\text{min})) : \text{LoopFragment} \rrbracket_{LF}
\end{aligned}$$

$$\begin{aligned}
& \llbracket \text{loopFrag} : \text{LoopFragment} \rrbracket_G = \\
& \quad \text{getCount.ID_LOOP.id}(\text{loopFrag})?count \rightarrow (\\
& \quad \text{not}(\text{condition}) \& \\
& \quad (\text{setCount.ID_LOOP.id}(\text{loopFrag})!0 \rightarrow \\
& \quad \text{guard.ID_LOOP.id}(\text{loopFrag}).1!false \rightarrow \textit{Skip}) \\
& \quad \square \\
& \quad (\text{count} == \text{defaultMax}) \& \\
& \quad (\text{setCount.ID_LOOP.id}(\text{loopFrag})!0 \rightarrow \\
& \quad \text{guard.ID_LOOP.id}(\text{loopFrag}).1!false \rightarrow \textit{Skip}) \\
& \quad \square \\
& \quad (\text{not}(\text{count} \geq \text{defaultMax}) \text{ and } \text{condition} == \text{true}) \& \\
& \quad (\text{setCount.ID_LOOP.id}(\text{loopFrag})!(\text{count} + 1) \rightarrow \\
& \quad \text{guard.ID_LOOP.id}(\text{loopFrag}).1!true \rightarrow \textit{Skip})), \\
& \text{if } \text{loopFrag} \in \llbracket (\text{loop}[\text{condition}]) : \text{LoopFragment} \rrbracket_{LF}
\end{aligned}$$

$$\begin{aligned}
& \llbracket \text{loopFrag} : \text{LoopFragment} \rrbracket_G = \\
& \quad \text{getCount.ID_LOOP.id}(\text{loopFrag})?count \rightarrow (\\
& \quad \text{not}(\text{condition}) \& \\
& \quad (\text{setCount.ID_LOOP.id}(\text{loopFrag})!0 \rightarrow \\
& \quad \text{guard.ID_LOOP.id}(\text{loopFrag}).1!false \rightarrow \textit{Skip}) \\
& \quad \square \\
& \quad ((\text{count} < \text{min}) \text{ and } \text{condition}) \& \\
& \quad (\text{setCount.ID_LOOP.id}(\text{loopFrag})!(\text{count} + 1) \rightarrow \\
& \quad \text{guard.ID_LOOP.id}(\text{loopFrag}).1!true \rightarrow \textit{Skip}) \\
& \quad \square \\
& \quad (\text{count} == \text{defaultMax}) \& \\
& \quad (\text{setCount.ID_LOOP.id}(\text{loopFrag})!0 \rightarrow \\
& \quad \text{guard.ID_LOOP.id}(\text{loopFrag}).1!false \rightarrow \textit{Skip}) \\
& \quad \square \\
& \quad (\text{not}(\text{count} < \text{min}) \text{ and } \text{not}(\text{count} \geq \text{defaultMax}) \text{ and } \text{condition}) \& \\
& \quad (\text{setCount.ID_LOOP.id}(\text{loopFrag})!(\text{count} + 1) \rightarrow \\
& \quad \text{guard.ID_LOOP.id}(\text{loopFrag}).1!true \rightarrow \textit{Skip}) \\
& \quad \square \\
& \quad ((\text{count} \geq \text{min}) \text{ and } (\text{count} < \text{defaultMax}) \text{ and } \text{condition}) \& \\
& \quad ((\text{setCount.ID_LOOP.id}(\text{loopFrag})!(\text{count} + 1) \rightarrow \\
& \quad \text{guard.ID_LOOP.id}(\text{loopFrag}).1!true \rightarrow \textit{Skip}) \sqcap \\
& \quad (\text{setCount.ID_LOOP.id}(\text{loopFrag})!0 \rightarrow \\
& \quad \text{guard.ID_LOOP.id}(\text{loopFrag}).1!false \rightarrow \textit{Skip}))), \\
& \text{if } \text{loopFrag} \in \llbracket (\text{loop}(\text{min})[\text{condition}]) : \text{LoopFragment} \rrbracket_{LF}
\end{aligned}$$

$$\begin{aligned}
& \llbracket \text{loopFrag} : \text{LoopFragment} \rrbracket_G = \\
& \quad \text{getCount.ID_LOOP.id}(\text{loopFrag})?count \rightarrow (\\
& \quad \quad (count < \text{min}) \& \\
& \quad \quad (\text{setCount.ID_LOOP.id}(\text{loopFrag})!(count + 1) \rightarrow \\
& \quad \quad \text{guard.ID_LOOP.id}(\text{loopFrag}).1!true \rightarrow \text{Skip}) \\
& \quad \quad \square \\
& \quad \quad (count == \text{max}) \& \\
& \quad \quad (\text{setCount.ID_LOOP.id}(\text{loopFrag})!0 \rightarrow \\
& \quad \quad \text{guard.ID_LOOP.id}(\text{loopFrag}).1!false \rightarrow \text{Skip}) \\
& \quad \quad \square \\
& \quad \quad (\text{not}(count < \text{min}) \text{ and } \text{not}(count \geq \text{max})) \& \\
& \quad \quad (\text{setCount.ID_LOOP.id}(\text{loopFrag})!(count + 1) \rightarrow \\
& \quad \quad \text{guard.ID_LOOP.id}(\text{loopFrag}).1!true \rightarrow \text{Skip}) \\
& \quad \quad \square \\
& \quad \quad (count \geq \text{min} \text{ and } count < \text{max}) \& \\
& \quad \quad ((\text{setCount.ID_LOOP.id}(\text{loopFrag})!(count + 1) \rightarrow \\
& \quad \quad \text{guard.ID_LOOP.id}(\text{loopFrag}).1!true \rightarrow \text{Skip}) \sqcap \\
& \quad \quad (\text{setCount.ID_LOOP.id}(\text{loopFrag})!0 \rightarrow \\
& \quad \quad \text{guard.ID_LOOP.id}(\text{loopFrag}).1!false \rightarrow \text{Skip}))), \\
& \text{if } \text{loopFrag} \in \llbracket (\text{loop}(\text{min}, \text{max})) : \text{LoopFragment} \rrbracket_{LF}
\end{aligned}$$

$$\begin{aligned}
& \llbracket \text{loopFrag} : \text{LoopFragment} \rrbracket_G = \\
& \quad \text{getCount.ID_LOOP.id}(\text{loopFrag})?count \rightarrow (\\
& \quad \quad (count < \text{defaultMin}) \& \\
& \quad \quad (\text{setCount.ID_LOOP.id}(\text{loopFrag})!(count + 1) \rightarrow \\
& \quad \quad \text{guard.ID_LOOP.id}(\text{loopFrag}).1!true \rightarrow \text{Skip}) \\
& \quad \quad \square \\
& \quad \quad (count == \text{defaultMax}) \& \\
& \quad \quad (\text{setCount.ID_LOOP.id}(\text{loopFrag})!0 \rightarrow \\
& \quad \quad \text{guard.ID_LOOP.id}(\text{loopFrag}).1!false \rightarrow \text{Skip}) \\
& \quad \quad \square \\
& \quad \quad (\text{not}(count < \text{defaultMin}) \text{ and } \text{not}(count \geq \text{defaultMax})) \& \\
& \quad \quad (\text{setCount.ID_LOOP.id}(\text{loopFrag})!(count + 1) \rightarrow \\
& \quad \quad \text{guard.ID_LOOP.id}(\text{loopFrag}).1!true \rightarrow \text{Skip}) \\
& \quad \quad \square \\
& \quad \quad (count \geq \text{defaultMin} \text{ and } count < \text{defaultMax}) \& \\
& \quad \quad (\text{setCount.ID_LOOP.id}(\text{loopFrag})!(count + 1) \rightarrow \\
& \quad \quad \text{guard.ID_LOOP.id}(\text{loopFrag}).1!true \rightarrow \text{Skip})), \\
& \text{if } \text{loopFrag} \in \llbracket (\text{loop}) : \text{LoopFragment} \rrbracket_{LF}
\end{aligned}$$

$$\begin{aligned}
& \llbracket \text{optFrag} : \text{OptFragment} \rrbracket_G = \\
& \quad (g_{-1_0} \& g_{-1_1} \& \dots \& g_{-1_n}) \& (guard.ID_OPT.id(\text{optFrag}).1!true) \\
& \quad \square \\
& \quad not((g_{-1_0} \& g_{-1_1} \& \dots \& g_{-1_n})) \& \\
& \quad (guard.ID_OPT.id(\text{optFrag}).1!false) \\
& \text{if } \text{optFrag} \in \llbracket (opt \ [g_0 \& g_1 \& \dots \& g_n] \ x_1 \ \text{end}) : \text{OptFragment} \rrbracket_{OF}
\end{aligned}$$

$$\begin{aligned}
& \llbracket \text{altFrag} : \text{AltFragment} \rrbracket_G = \\
& \quad (guard.ID_ALT.id(\text{altFrag}).1?true \rightarrow guard.ID_ALT.id(\text{altFrag}).2?false \rightarrow \\
& \quad \dots \rightarrow guard.ID_ALT.id(\text{altFrag}).n!false) \\
& \quad \square \\
& \quad (guard.ID_ALT.id(\text{altFrag}).1?false \rightarrow guard.ID_ALT.id(\text{altFrag}).2?true \rightarrow \\
& \quad \dots \rightarrow guard.ID_ALT.id(\text{altFrag}).n!false) \\
& \quad \square \\
& \quad \dots \\
& \quad \square \\
& \quad (guard.ID_ALT.id(\text{altFrag}).1?false \rightarrow guard.ID_ALT.id(\text{altFrag}).2?false \rightarrow \\
& \quad \dots \rightarrow guard.ID_ALT.id(\text{altFrag}).n!true) \\
& \quad \square \\
& \quad (guard.ID_ALT.id(\text{altFrag}).1?false \rightarrow guard.ID_ALT.id(\text{altFrag}).2?false \rightarrow \\
& \quad \dots \rightarrow guard.ID_ALT.id(\text{altFrag}).n!false), \\
& \text{if } \text{altFrag} \in \llbracket (\text{alt } x_1 \ \text{else } x_2 \ \text{else } \dots \ x_n \ \text{end}) : \text{AltFragment} \rrbracket_{AF}
\end{aligned}$$

$$\begin{aligned}
& \llbracket \text{altFrag} : \text{AltFragment} \rrbracket_G = \\
& \quad (g_0 \& g_1 \& \dots \& g_n) \& (guard.ID_ALT.id(\text{altFrag}).1!true) \\
& \quad \square \\
& \quad not((g_0 \& g_1 \& \dots \& g_n)) \& \\
& \quad (guard.ID_ALT.id(\llbracket \text{altFrag} \rrbracket_{AF}).1!false), \\
& \text{if } \text{altFrag} \in \llbracket (\text{alt } [g_0 \& g_1 \& \dots \& g_n] \ x_1 \ \text{end}) : \text{AltFragment} \rrbracket_{AF}
\end{aligned}$$

$$\begin{aligned}
& \llbracket \text{altFrag} : \text{AltFragment} \rrbracket_G = \\
& \quad (g_{-1_0} \& g_{-1_1} \& \dots \& g_{-1_n}) \& (guard.ID_ALT.id(\text{altFrag}).1!true) \\
& \quad \square \\
& \quad not((g_{-1_0} \& g_{-1_1} \& \dots \& g_{-1_n})) \& \\
& \quad (guard.ID_ALT.id(\text{altFrag}).1!false), \\
& \text{if } \text{altFrag} \in \llbracket (\text{alt } [g_{-1_0} \& g_{-1_1} \& \dots \& g_{-1_n}] \ x_1 \ \text{else } x_2 \ \text{end}) : \text{AltFragment} \rrbracket_{AF}
\end{aligned}$$

$$\begin{aligned}
\llbracket \text{altFrag} : \text{AltFragment} \rrbracket_G = & \\
& (g_{1_0} \& g_{1_1} \& \dots \& g_{1_n}) \& (\text{guard.ID_ALT.id}(\text{altFrag}).1!true \rightarrow \dots \\
& \rightarrow \text{guard.ID_ALT.id}(\text{altFrag}).n!false) \\
& \square \\
& \dots \\
& \square \\
& (g_{n_0} \& g_{n_1} \& \dots \& g_{n_n}) \& (\text{guard.ID_ALT.id}(\text{altFrag}).1!false \rightarrow \dots \\
& \rightarrow \text{guard.ID_ALT.id}(\text{altFrag}).n!true) \\
& \square \\
& \text{not}((g_{1_0} \& g_{1_1} \& \dots \& g_{1_n}) \text{ or } \dots \text{ or } (g_{n_0} \& g_{n_1} \& \dots \& g_{n_n})) \& \\
& (\text{guard.ID_ALT.id}(\text{altFrag}).1!false \rightarrow \dots \\
& \rightarrow \text{guard.ID_ALT.id}(\text{altFrag}).n!true), \\
\text{if } \text{altFrag} \in & \llbracket (\text{alt } [g_{1_0} \& g_{1_1} \& \dots \& g_{1_n}] x_1 \text{ else } \dots \text{ else} \\
& [g_{n_0} \& g_{n_1} \& \dots \& g_{n_n}] x_n \text{ end}) : \text{AltFragment} \rrbracket_{AF}
\end{aligned}$$

$$\begin{aligned}
\llbracket \text{altFrag} : \text{AltFragment} \rrbracket_G = & \\
& (g_{1_0} \& g_{1_1} \& \dots \& g_{1_n}) \& (\text{guard.ID_ALT.id}(\text{altFrag}).1!true \rightarrow \dots \\
& \rightarrow \text{guard.ID_ALT.id}(\text{altFrag}).n!false) \\
& \square \\
& \dots \\
& \square \\
& (g_{n_0} \& g_{n_1} \& \dots \& g_{n_n}) \& (\text{guard.ID_ALT.id}(\text{altFrag}).1!false \rightarrow \dots \\
& \rightarrow \text{guard.ID_ALT.id}(\text{altFrag}).n!true) \\
& \square \\
& \text{not}((g_{1_0} \& g_{1_1} \& \dots \& g_{1_n}) \text{ or } \dots \text{ or } (g_{n_0} \& g_{n_1} \& \dots \& g_{n_n})) \& \\
& (\text{guard.ID_ALT.id}(\text{altFrag}).1!false \rightarrow \dots \rightarrow \\
& \text{guard.ID_ALT.id}(\text{altFrag}).n!true), \\
\text{if } \text{altFrag} \in & \llbracket (\text{alt } [g_{1_0} \& g_{1_1} \& \dots \& g_{1_n}] x_1 \text{ else } \dots \text{ else} \\
& [g_{n_0} \& g_{n_1} \& \dots \& g_{n_n}] x_n \text{ else } x_m \text{ end}) : \text{AltFragment} \rrbracket_{AF}
\end{aligned}$$

$$\begin{aligned}
\llbracket \text{loopFrag} : \text{LoopFragment} \rrbracket_{LF} = & \\
& \text{loop.ID_LOOP.id}(\text{loopFrag}) \rightarrow \text{guard.ID_LOOP.id}(\text{loopFrag}).1?id1 \rightarrow \\
& (id1 \& (\llbracket x_1 \rrbracket_P) \square \text{not}(id1) \& (\mathbf{Skip})) \\
\text{if } \text{loopFrag} \in & \llbracket \text{LoopFragment} \rrbracket_{LF}
\end{aligned}$$

$$\begin{aligned}
\llbracket \text{optFrag} : \text{OptFragment} \rrbracket_{OF} = & \\
& \text{opt.ID_OPT.id}(\text{optFrag}) \rightarrow \text{guard.ID_OPT.id}(\text{optFrag}).1?id1 \rightarrow \\
& (id1 \& (\llbracket x_1 \rrbracket_P) \square \text{not}(id1) \& (\mathbf{Skip})) \\
\text{if } \text{optFrag} \in & \llbracket \text{OptFragment} \rrbracket_{OF}
\end{aligned}$$

$$\begin{aligned}
& \llbracket \text{altFrag} : \text{AltFragment} \rrbracket_{AF} = \\
& \quad \text{alt.ID_ALT}\{id\} \rightarrow \text{guard.ID_ALT}\{id\}_1?id1 \rightarrow \\
& \quad \quad \text{guard.ID_ALT}\{id\}_2?id2 \rightarrow \dots \text{guard.ID_ALT}\{id\}_n?idn \\
& \quad \rightarrow (id1 \& (\llbracket x_1 \rrbracket_P) \sqcap id2 \& (\llbracket x_2 \rrbracket_P) \sqcap \dots idn \& (\llbracket x_n \rrbracket_P)) \\
& \text{if } \text{altFrag} \in \llbracket (\text{alt } x_1 \text{ else } x_2 \text{ else } \dots x_n \text{ end}) : \text{AltFragment} \rrbracket_{AF}
\end{aligned}$$

$$\begin{aligned}
& \llbracket \text{altFrag} : \text{OptFragment} \rrbracket_{AF} = \\
& \quad \text{alt.ID_ALT}\{id\} \rightarrow \text{guard.ID_ALT}\{id\}.1?id1 \\
& \quad \rightarrow (id1 \& (\llbracket x_1 \rrbracket_P) \sqcap \text{not}(id1) \& (\mathbf{Skip})) \\
& \text{if } \text{altFrag} \in \llbracket (\text{alt } [g_0 \& g_1 \& \dots \& g_n] x_1 \text{ end}) : \text{OptFragment} \rrbracket_{AF}
\end{aligned}$$

$$\begin{aligned}
& \llbracket \text{altFrag} : \text{AltFragment} \rrbracket_{AF} = \\
& \quad \text{alt.ID_ALT}\{id\} \rightarrow \text{guard.ID_ALT}\{id\}.1?id1 \\
& \quad \rightarrow (id1 \& (\llbracket x_1 \rrbracket_P) \sqcap \text{not}(id1) \& (\llbracket x_2 \rrbracket_P)) \\
& \text{if } \text{altFrag} \in \llbracket (\text{alt } [g_1_0 \& g_1_1 \& \dots \& g_1_n] x_1 \text{ else } x_2 \text{ end}) : \text{AltFragment} \rrbracket_{AF}
\end{aligned}$$

$$\begin{aligned}
& \llbracket \text{altFrag} : \text{AltFragment} \rrbracket_{AF} = \\
& \quad \text{alt.ID_ALT}\{id\} \rightarrow \text{guard.ID_ALT}\{id\}.1?id1 \rightarrow \dots \\
& \quad \rightarrow \text{guard.ID_ALT}\{id\}.n?idn \\
& \quad \rightarrow (id1 \& (\llbracket x_1 \rrbracket_P) \sqcap \dots idn \& (\llbracket x_n \rrbracket_P)) \\
& \quad \sqcap \text{not}(id1 \text{ or } \dots \text{ or } idn) \& (\mathbf{Skip})) \\
& \text{if } \text{altFrag} \in \llbracket (\text{alt } [g_1_0 \& g_1_1 \& \dots \& g_1_n] x_1 \text{ else } \dots \text{ else } \\
& \quad [g_n_0 \& g_n_1 \& \dots \& g_n_n] x_n \text{ end}) : \text{AltFragment} \rrbracket_{AF}
\end{aligned}$$

$$\begin{aligned}
& \llbracket \text{altFrag} : \text{AltFragment} \rrbracket_{AF} = \\
& \quad \text{alt.ID_ALT}\{id\} \rightarrow \text{guard.ID_ALT}\{id\}.1?id1 \rightarrow \dots \\
& \quad \rightarrow \text{guard.ID_ALT}\{id\}.n?idn \\
& \quad \rightarrow (id1 \& (\llbracket x_1 \rrbracket_P) \sqcap \dots idn \& (\llbracket x_n \rrbracket_P)) \\
& \quad \sqcap \text{not}(id1 \text{ or } \dots \text{ or } idn) \& (\llbracket x_m \rrbracket_P)) \\
& \text{if } \text{altFrag} \in \llbracket (\text{alt } [g_1_0 \& g_1_1 \& \dots \& g_1_n] x_1 \text{ else } \dots \text{ else } \\
& \quad [g_n_0 \& g_n_1 \& \dots \& g_n_n] x_n \text{ else } x_m \text{ end}) : \text{AltFragment} \rrbracket_{AF}
\end{aligned}$$

$$\begin{aligned}
\text{parallel}(\text{parFrag}) &= \square \text{parFrag} : \text{parFrag} \bullet \\
&\quad (\text{par.ID_PAR}.\text{id}(\llbracket \text{parFrag} : \text{ParFragment} \rrbracket_{\text{PF}}) \rightarrow \\
&\quad \parallel \text{thread} : \text{parFrag}.\text{threads} \bullet \text{parallel}(\text{id}(\llbracket \text{parFrag} \rrbracket_{\text{PF}}), \text{id}(\text{varstthread}))) \\
\text{parallel}(\text{id}, \text{threadID}) &= \parallel \text{actor} : \llbracket \text{parFrag} \rrbracket_{\text{PF}}.\text{threadID}.\text{actors} \bullet \\
&\quad \text{parallel}(\text{actor}, \text{id}, \text{threadID}) \\
\text{parallel}(\text{a}, \text{id}, \text{threadID}) &= \S \text{frag} : \llbracket \text{parFrag} \rrbracket_{\text{PF}}.\text{threadID}.\text{interactionFragments} \bullet \llbracket \text{frag} \rrbracket_{(\text{a})}^F
\end{aligned}$$

$$\begin{aligned}
\text{strict}(\text{strFrag}) &= \square \text{strFrag} : \text{strFrag} \bullet \\
&\quad (\text{str.ID_STR}.\text{id}(\llbracket \text{strFrag} : \text{StrFragment} \rrbracket_{\text{SF}}) \rightarrow \text{strict}(\text{id}(\llbracket \text{strFrag} \rrbracket_{\text{SF}}))) \\
\text{strict}(\text{id}) &= \S \text{interactionFrag} : \llbracket \text{strFrag} \rrbracket_{\text{SF}}.\text{interactionFragments}
\end{aligned}$$

$$\begin{aligned}
&\llbracket (\text{a}_1 -> \text{a}_2 : \text{mID!var}) : \text{InteractionFragment} \rrbracket_F \hat{=} \\
&\quad \text{a}_1 = \text{mID!var} \rightarrow \textit{Skip} \\
&\quad \text{a}_2 = \text{mID?var} \rightarrow \textit{set_var!var} \rightarrow \textit{Skip} \\
&\llbracket (\text{a}_1 -> \text{a}_2 : \text{mID?var}) : \text{InteractionFragment} \rrbracket_F \hat{=} \\
&\quad \text{a}_1 = \text{mID?var} \rightarrow \textit{set_var!var} \rightarrow \textit{Skip} \\
&\quad \text{a}_2 = \text{mID!var} \rightarrow \textit{Skip} \\
&\llbracket (\text{a}_1 -> \text{a}_1 : \text{mID?var}) : \text{InteractionFragment} \rrbracket_F \hat{=} \\
&\quad \text{a}_1 = \text{mID?var} \rightarrow \textit{set_var!var} \rightarrow \textit{Skip} \\
&\llbracket (\text{a}_1 -> \text{a}_2 : \text{mID}) : \text{InteractionFragment} \rrbracket_F \hat{=} \\
&\quad \text{a}_1 = \text{mID} \rightarrow \textit{Skip} \\
&\quad \text{a}_2 = \text{mID} \rightarrow \textit{Skip}
\end{aligned}$$

Key

$\llbracket D \rrbracket$	\rightarrow RoboHumansDocument
$\llbracket CS \rrbracket$	\rightarrow Capabilities
$\llbracket SDS \rrbracket$	\rightarrow SequenceDiagramGroup
$\llbracket SD \rrbracket$	\rightarrow <i>An instance of SequenceDiagramGroup</i>
$\llbracket CO \rrbracket$	\rightarrow Constants
$\llbracket LS \rrbracket$	\rightarrow LifelineBlock
$\llbracket TU \rrbracket$	\rightarrow TimeUnit
$\llbracket CONS \rrbracket$	\rightarrow ConstAssignment
$\llbracket CON \rrbracket$	\rightarrow <i>An instance of ConstAssignment</i>
$\llbracket OS \rrbracket$	\rightarrow Out
$\llbracket O \rrbracket$	\rightarrow <i>An instance of Out</i>
$\llbracket IS \rrbracket$	\rightarrow In
$\llbracket I \rrbracket$	\rightarrow <i>An instance of In</i>
$\llbracket VS \rrbracket$	\rightarrow VarsList
$\llbracket V \rrbracket$	\rightarrow <i>An instance of VarsList</i>
$\llbracket TO \rrbracket$	\rightarrow <i>An Out LifelineEvent instance with a type</i>
$\llbracket TI \rrbracket$	\rightarrow <i>An In LifelineEvent instance with a type</i>
$\llbracket SS \rrbracket$	\rightarrow Sequence
$\llbracket S \rrbracket$	\rightarrow <i>An instance of Sequence</i>
$\llbracket A \rrbracket$	\rightarrow Actor
$\llbracket FS \rrbracket$	\rightarrow InteractionFragment
$\llbracket F \rrbracket$	\rightarrow <i>An instance of InteractionFragment</i>
$\llbracket F(a) \rrbracket$	\rightarrow <i>An instance of InteractionFragment for Actor a</i>
$\llbracket G \rrbracket$	\rightarrow <i>The body of the evaluation for an InteractionFragment with a Guard</i>
$\llbracket AF \rrbracket$	\rightarrow <i>An AltFragment</i>
$\llbracket OF \rrbracket$	\rightarrow <i>An OptFragment</i>
$\llbracket LF \rrbracket$	\rightarrow <i>A LoopFragment</i>
$\llbracket PF \rrbracket$	\rightarrow <i>A ParFragment</i>
$\llbracket SF \rrbracket$	\rightarrow <i>A StrFragment</i>
$\llbracket P \rrbracket$	\rightarrow <i>An operand inside an InteractionFragment with a Guard</i>