section *stateflow_toolkit* **parents** *circus_toolkit*, *basic_toolkit*

function 40 **leftassoc** $(\_ \wedge_{\mathcal{A}} \_)$
function 40 **leftassoc** $(\_ \vee_{\mathcal{A}} \_)$
function$(\neg_{\mathcal{A}} \_)$
function 40 **leftassoc** $(\_ \wedge_{\mathcal{B}} \_)$
function 40 **leftassoc** $(\_ \vee_{\mathcal{B}} \_)$
function 40 **leftassoc** $(\_ \oplus_{\mathcal{B}} \_)$
function$(\neg_{\mathcal{B}} \_)$
function 40 **leftassoc** $(\_ << \_)$
function 40 **leftassoc** $(\_ >> \_)$
function 40 **leftassoc** $(\_ <_{\mathcal{A}} \_)$
function 40 **leftassoc** $(\_ >_{\mathcal{A}} \_)$
function 40 **leftassoc** $(\_ \leq_{\mathcal{A}} \_)$
function 40 **leftassoc** $(\_ \geq_{\mathcal{A}} \_)$
function 40 **leftassoc** $(\_ =_{\mathcal{A}} \_)$
function 40 **leftassoc** $(\_ \neq_{\mathcal{A}} \_)$

$$\_ \wedge_{\mathcal{A}} \_ : \mathbb{A} \times \mathbb{A} \to \mathbb{A}$$
$$\_ \vee_{\mathcal{A}} \_ : \mathbb{A} \times \mathbb{A} \to \mathbb{A}$$
$$\neg_{\mathcal{A}} \_ : \mathbb{A} \to \mathbb{A}$$
$$\_ \wedge_{\mathcal{B}} \_ : \mathbb{A} \times \mathbb{A} \to \mathbb{A}$$
$$\_ \vee_{\mathcal{B}} \_ : \mathbb{A} \times \mathbb{A} \to \mathbb{A}$$
$$\_ \oplus_{\mathcal{B}} \_ : \mathbb{A} \times \mathbb{A} \to \mathbb{A}$$
$$\neg_{\mathcal{B}} \_ : \mathbb{A} \to \mathbb{A}$$
$$\_ << \_ : \mathbb{A} \times \mathbb{A} \to \mathbb{A}$$
$$\_ >> \_ : \mathbb{A} \times \mathbb{A} \to \mathbb{A}$$
$$\_ <_{\mathcal{A}} \_ : \mathbb{A} \times \mathbb{A} \to \mathbb{A}$$
$$\_ >_{\mathcal{A}} \_ : \mathbb{A} \times \mathbb{A} \to \mathbb{A}$$
$$\_ \leq_{\mathcal{A}} \_ : \mathbb{A} \times \mathbb{A} \to \mathbb{A}$$
$$\_ \geq_{\mathcal{A}} \_ : \mathbb{A} \times \mathbb{A} \to \mathbb{A}$$
$$\_ =_{\mathcal{A}} \_ : \mathbb{A} \times \mathbb{A} \to \mathbb{A}$$
$$\_ \neq_{\mathcal{A}} \_ : \mathbb{A} \times \mathbb{A} \to \mathbb{A}$$

$$\forall x, y : \mathbb{A} \bullet x \wedge_{\mathcal{A}} y = \neg_{\mathcal{A}} (\neg_{\mathcal{A}} x \vee_{\mathcal{A}} \neg_{\mathcal{A}} y)$$
$$\forall x, y : \mathbb{A} \bullet x \wedge_{\mathcal{B}} y = \neg_{\mathcal{B}} (\neg_{\mathcal{B}} x \vee_{\mathcal{B}} \neg_{\mathcal{B}} y)$$
$$\forall x, y : \mathbb{A} \bullet x \oplus_{\mathcal{B}} y = (x \wedge_{\mathcal{B}} \neg_{\mathcal{B}} y) \vee_{\mathcal{B}} (\neg_{\mathcal{B}} x \wedge_{\mathcal{B}} y)$$

$DECOMPOSITION ::= SET \mid CLUSTER$
$TYPE ::= AND \mid OR \mid CHART$
$DESTINATION ::= STATE \mid JUNCTION$
$[SID, TID, JID, EVENT]$
$NID ::= snode\langle\langle SID \rangle\rangle \mid jnode\langle\langle JID \rangle\rangle$
$SFBOOL == \mathbb{Z}$

State

$$identifier : SID$$
$$default, inner, outer : TID$$
$$parent, left, right : SID$$
$$substates : \mathrm{seq}\ SID$$
$$decomposition : DECOMPOSITION$$
$$type : TYPE$$
$$history : \mathbb{B}$$

Junction

$identifier : JID$
$transition : TID$
$parent : SID$
$history : \mathbb{B}$

Transition

$identifier : TID$
$source, destination : NID$
$next : TID$
$parent : SID$

$nullstate : State$
$nulljunction : Junction$
$nulltransition : Transition$

$nullstate.history = \textbf{False}$
$nulljunction.history = \textbf{False}$

StateflowChart

$identifier : SID$
$states : SID \pfun State$
$transitions : TID \pfun Transition$
$junctions : JID \pfun Junction$

$nullstate \notin \operatorname{ran} states$
$nulltransition \notin \operatorname{ran} transitions$
$nulljunction \notin \operatorname{ran} junctions$
$\#\{s : \operatorname{ran} states \mid s.type = CHART\} = 1$
$(states(identifier)).type = CHART$
$\forall n : SID \mid n \in \operatorname{dom} states \bullet (states(n)).identifier = n$
$\forall n : JID \mid n \in \operatorname{dom} junctions \bullet (junctions(n)).identifier = n$
$\forall n : TID \mid n \in \operatorname{dom} transitions \bullet (transitions(n)).identifier = n$

$parent : State \leftrightarrow State$

$\forall s_1, s_2 : State \bullet parent(s_1) = s_2 \Leftrightarrow s_1.parent = s_2.identifier$

$ancestors : State \to \mathbb{P}\, State$

$\forall s : State \bullet ancestors(s) = (parent^{+})\,(\!|\,\{s\}\,|\!)\setminus\{nullstate\}$

$la : (State \times State) \to State$

$\forall s_1, s_2 : State \bullet la(s_1, s_2) = \mu\, x : (ancestors(s_1) \cap ancestors(s_2)) \mid$
$\quad (\forall y : (ancestors(s_1) \cap ancestors(s_2)) \bullet x = y \lor y \in ancestors(x)) \bullet x$

channel *read_inputs*, *write_outputs*
channel *input_event* : $\operatorname{seq} \mathbb{B}$
channel *events* : $\operatorname{seq} EVENT$
channel *local_event* : $EVENT \times State$
channel *executeentryaction*, *executeexitaction* : SID
channel *executeduringaction* : $SID \times EVENT$
channel *executeconditionaction*, *executetransitionaction* : TID
channel *evaluatecondition* : $TID \times \mathbb{B}$
channel *checktrigger* : $TID \times EVENT$
channel *result* : $TID \times EVENT \times \mathbb{B}$

channel $activate, deactivate : SID$
channel $junction : JID \times Junction$
channel $transition : TID \times Transition$
channel $state : SID \times State$
channel $chart : State$
channel $status : SID \times \mathbb{B}$
channel $history : SID \times SID$
channel $end\_local\_execution, end\_action$
channel $interrupt : \mathbb{B}$
channel end_cycle
channelset $interface == \{\!|\ executeentryaction, executeduringaction, executeexitaction, executeconditionaction, executetransitionaction, evaluatecondition, checktrigger, result, junction, transition, state, chart, status, history, activate, deactivate, read\_inputs, write\_outputs, end\_local\_execution, end\_action, local\_event, interrupt, events\ |\!\}$

$and, or : \mathbb{B} \times \mathbb{B} \to \mathbb{B}$
$not : \mathbb{B} \to \mathbb{B}$

$or(\mathbf{False}, \mathbf{False}) = and(\mathbf{False}, \mathbf{False}) = and(\mathbf{True}, \mathbf{False}) = and(\mathbf{False}, \mathbf{True}) = \mathbf{False}$
$and(\mathbf{True}, \mathbf{True}) = or(\mathbf{True}, \mathbf{True}) = or(\mathbf{True}, \mathbf{False}) = or(\mathbf{False}, \mathbf{True}) = \mathbf{True}$
$not(\mathbf{True}) = \mathbf{False}$
$not(\mathbf{False}) = \mathbf{True}$

process $Simulator \mathrel{\widehat=} \mathbf{begin}$

$entryActionCheck \mathrel{\widehat=} \mathbf{val}\ sid : SID;\ \mathbf{vres}\ b : \mathbb{B} \bullet status!sid?active \longrightarrow b := not(active)$
$duringActionCheck \mathrel{\widehat=} \mathbf{val}\ sid : SID;\ \mathbf{vres}\ b : \mathbb{B} \bullet entryActionCheck(sid, b)$
$exitActionCheck \mathrel{\widehat=} \mathbf{val}\ sid : SID;\ \mathbf{vres}\ b : \mathbb{B} \bullet entryActionCheck(sid, b)$
$conditionActionCheck \mathrel{\widehat=} \mathbf{val}\ sid : SID;\ \mathbf{vres}\ b : \mathbb{B} \bullet entryActionCheck(sid, b)$
$transitionActionCheck \mathrel{\widehat=} \mathbf{val}\ sid : SID;\ \mathbf{vres}\ b : \mathbb{B} \bullet$
$entryActionCheck(sid, b)\ ;\ \mathbf{var}\ ss : \mathrm{seq}\ SID \bullet$
$$\left(\begin{array}{l} state!sid?s \longrightarrow ss := s.substates; \\ \mu X \bullet \left(\begin{array}{l} \mathbf{if}\ ss = \langle\rangle \longrightarrow \mathbf{Skip} \\ [\!]\ ss \neq \langle\rangle \longrightarrow \left(\begin{array}{l} status!(head\ ss)?active \longrightarrow b := or(b, active); \\ ss := tail\ ss\ ;\ X \end{array} \right) \\ \mathbf{fi} \end{array} \right) \end{array} \right)$$

$exitStatesCheck \mathrel{\widehat=} \mathbf{val}\ sid : SID;\ \mathbf{vres}\ b : \mathbb{B} \bullet transitionActionCheck(sid, b)$
$enterState1Check \mathrel{\widehat=} \mathbf{val}\ sid : SID;\ \mathbf{vres}\ b : \mathbb{B} \bullet$
$state!sid?s \longrightarrow entryActionCheck(s.parent, b)$
$enterState2Check \mathrel{\widehat=} \mathbf{val}\ sid : SID;\ \mathbf{vres}\ b : \mathbb{B} \bullet state!sid?s \longrightarrow$
$$\left(\begin{array}{l} \mathbf{if}\ s.left = nullstate.identifier \longrightarrow b := \mathbf{False} \\ [\!]\ s.left \neq nullstate.identifier \longrightarrow entryActionCheck(s.left, b) \\ \mathbf{fi} \end{array} \right)$$
$executeStateCheck \mathrel{\widehat=} \mathbf{val}\ sid : SID;\ \mathbf{vres}\ b : \mathbb{B} \bullet$
$state!sid?s \longrightarrow entryActionCheck(s.parent, b)$
$exitStateCheck \mathrel{\widehat=} \mathbf{val}\ sid : SID;\ \mathbf{vres}\ b : \mathbb{B} \bullet state!sid?s \longrightarrow entryActionCheck(s.parent, b)$
$enterState15Check \mathrel{\widehat=} \mathbf{val}\ sid : SID;\ \mathbf{vres}\ b : \mathbb{B} \bullet entryActionCheck(sid, b)$

$ExecuteTransition \mathrel{\widehat=}$
$\mathbf{val}\ tid : TID;\ \mathbf{val}\ path : \mathrm{seq}\ TID;\ \mathbf{val}\ source : State;\ \mathbf{val}\ ce : EVENT;\ \mathbf{vres}\ success : \mathbb{B} \bullet$
$$\left(\begin{array}{l}
\mathbf{if}\ tid = nulltransition.identifier \longrightarrow \\
\left(\begin{array}{l}
\mathbf{if}\ path = \varnothing \longrightarrow success := \mathbf{False} \\
[\!]\ path \neq \varnothing \longrightarrow \left(\begin{array}{l} transition!(last\ path)?lt \longrightarrow \\ ExecuteTransition(lt.next, (front\ path), source, ce, success) \end{array}\right) \\
\mathbf{fi}
\end{array}\right) \\
[\!]\ tid \neq nulltransition.identifier \longrightarrow \\
CheckValidity(tid, path, source, ce, success) \\
\mathbf{fi}
\end{array}\right)$$

$CheckValidity \mathrel{\widehat=}$
$\mathbf{val}\ tid : TID;\ \mathbf{val}\ path : \mathrm{seq}\ TID;\ \mathbf{val}\ source : State;\ \mathbf{val}\ ce : EVENT;\ \mathbf{vres}\ success : \mathbb{B} \bullet$
$checktrigger!tid!ce \longrightarrow result!tid!ce?e \longrightarrow evaluatecondition!tid?c \longrightarrow$
$$\left(\begin{array}{l}
\mathbf{if}\ e = \mathbf{True} \wedge c = \mathbf{True} \longrightarrow \\
\quad \left(\begin{array}{l}
executeconditionaction!tid \longrightarrow LocalEventCondition(source.identifier); \\
\mathbf{var}\ b : \mathbb{B} \bullet \\
\left(\begin{array}{l}
conditionActionCheck(source.identifier, b); \\
\left(\begin{array}{l}
\mathbf{if}\ b = \mathbf{True} \longrightarrow success := \mathbf{True} \\
[\!]\ b = \mathbf{False} \longrightarrow Proceed(tid, path \frown \langle tid \rangle, source, ce, success) \\
\mathbf{fi}
\end{array}\right)
\end{array}\right)
\end{array}\right) \\
[\!]\ \neg\ (e = \mathbf{True} \wedge c = \mathbf{True}) \longrightarrow \left(\begin{array}{l} transition!tid?t \longrightarrow \\ ExecuteTransition(t.next, path, source, ce, success) \end{array}\right) \\
\mathbf{fi}
\end{array}\right)$$

$Proceed \mathrel{\widehat=}$
$\mathbf{val}\ tid : TID;\ \mathbf{val}\ path : \mathrm{seq}\ TID;\ \mathbf{val}\ source : State;\ \mathbf{val}\ ce : EVENT;\ \mathbf{vres}\ success : \mathbb{B} \bullet$
$transition!tid?t \longrightarrow$
$$\left(\begin{array}{l}
\mathbf{if}\ t.destination \in \mathrm{ran}\ snode \longrightarrow \left(\begin{array}{l} state!((snode^{\sim})\ t.destination)?dest \longrightarrow \\ proceedToState(source, dest, path, ce, success) \end{array}\right) \\
[\!]\ t.destination \in \mathrm{ran}\ jnode \longrightarrow proceedToJunction(tid, path, source, ce, success) \\
\mathbf{fi}
\end{array}\right)$$

$proceedToState \mathrel{\widehat=}$
$\mathbf{val}\ src, dest : State;\ \mathbf{val}\ path : \mathrm{seq}\ TID;\ \mathbf{val}\ ce : EVENT;\ \mathbf{vres}\ success : \mathbb{B} \bullet$
$$\left(\begin{array}{l}
ExitStates((la(src, dest)).substates, ce); \\
\mathbf{var}\ b : \mathbb{B} \bullet \\
\left(\begin{array}{l}
exitStatesCheck((la(src, dest)).identifier, b); \\
\left(\begin{array}{l}
\mathbf{if}\ b = \mathbf{True} \longrightarrow \mathbf{Skip} \\
[\!]\ b = \mathbf{False} \longrightarrow \\
\quad \left(\begin{array}{l}
executePath(path, src, dest, ce); \\
transitionActionCheck((la(src, dest)).identifier, b); \\
\left(\begin{array}{l}
\mathbf{if}\ b = \mathbf{True} \longrightarrow \mathbf{Skip} \\
[\!]\ b = \mathbf{False} \longrightarrow EnterState(dest, la(src, dest), ce) \\
\mathbf{fi}
\end{array}\right)
\end{array}\right) \\
\mathbf{fi}
\end{array}\right)
\end{array}\right); \\
success := \mathbf{True}
\end{array}\right)$$

$$
\begin{array}{l}
executePath \mathrel{\widehat=} path : \mathrm{seq}\ TID;\ src, dest : State;\ ce : EVENT \bullet \\
\left(\begin{array}{l}
\textbf{if}\ \#path = 0 \longrightarrow \textbf{Skip} \\
\quad [\!]\ \#path > 0 \longrightarrow \left(\begin{array}{l}
executetransitionaction!(head\ path) \longrightarrow \\
LocalEventTransition((la(src, dest)).identifier); \\
\textbf{var}\ b : \mathbb{B} \bullet \\
\left(\begin{array}{l}
transitionActionCheck((la(src, dest)).identifier, b); \\
\left(\begin{array}{l}
\textbf{if}\ b = \textbf{True} \longrightarrow \textbf{Skip} \\
[\!]\ b = \textbf{False} \longrightarrow executePath(tail\ path, src, dest, ce) \\
\textbf{fi}
\end{array}\right)
\end{array}\right)
\end{array}\right) \\
\textbf{fi}
\end{array}\right)
\end{array}
$$

$$
\begin{array}{l}
proceedToJunction \mathrel{\widehat=} \\
\textbf{val}\ tid : TID;\ \textbf{val}\ path : \mathrm{seq}\ TID;\ \textbf{val}\ source : State;\ \textbf{val}\ ce : EVENT;\ \textbf{vres}\ success : \mathbb{B} \bullet \\
\quad transition!tid?t \longrightarrow junction!((jnode^{\sim})\ t.destination)?dj \longrightarrow \\
\quad \left(\begin{array}{l}
\textbf{if}\ dj.history = \textbf{False} \longrightarrow executeJunction(dj, path, source, ce, success) \\
[\!]\ dj.history = \textbf{True} \longrightarrow history!(dj.parent)?lsid \longrightarrow \\
\quad \left(\begin{array}{l}
\textbf{if}\ lsid = nullstate.identifier \longrightarrow \\
\quad \left(\begin{array}{l}
state!(dj.parent)?s \longrightarrow \\
ExecuteDefaultTransition(s, s, ce)\ ;\ success := \textbf{True}
\end{array}\right) \\
[\!]\ lsid \neq nullstate.identifier \longrightarrow \\
\quad \left(\begin{array}{l}
state!lsid?ls \longrightarrow \\
proceedToState(source, ls, path, ce, success)
\end{array}\right) \\
\textbf{fi}
\end{array}\right) \\
\textbf{fi}
\end{array}\right)
\end{array}
$$

$$
\begin{array}{l}
executeJunction \mathrel{\widehat=} \textbf{val}\ j : Junction;\ \textbf{val}\ path : \mathrm{seq}\ TID; \\
\textbf{val}\ source : State;\ \textbf{val}\ ce : EVENT;\ \textbf{vres}\ success : \mathbb{B} \bullet \\
\quad \left(\begin{array}{l}
\textbf{if}\ j.transition = nulltransition.identifier \longrightarrow \\
\quad success := \textbf{False} \\
[\!]\ j.transition \neq nulltransition.identifier \longrightarrow \\
\quad ExecuteTransition(j.transition, path, source, ce, success) \\
\textbf{fi}
\end{array}\right)
\end{array}
$$

$$
\begin{array}{l}
ExecuteDefaultTransition \mathrel{\widehat=} s, tpp : State;\ ce : EVENT \bullet \\
\quad \left(\begin{array}{l}
\textbf{if}\ s.default \neq nulltransition.identifier \longrightarrow \\
\quad \left(\begin{array}{l}
\textbf{var}\ success : \mathbb{B} \bullet ExecuteTransition(s.default, \langle\rangle, s, ce, success); \\
(\textbf{if}\ success = \textbf{True} \longrightarrow \textbf{Skip}\ [\!]\ success = \textbf{False} \longrightarrow \textbf{Stop}\ \textbf{fi})
\end{array}\right) \\
[\!]\ s.default = nulltransition.identifier \longrightarrow \\
\quad \left(\begin{array}{l}
\textbf{if}\ \#s.substates = 0 \longrightarrow \textbf{Skip} \\
[\!]\ \#s.substates = 1 \longrightarrow \left(\begin{array}{l}
state!(head\ s.substates)?saux \longrightarrow \\
EnterState(saux, tpp, ce)
\end{array}\right) \\
[\!]\ \#s.substates > 1 \longrightarrow \textbf{Stop} \\
\textbf{fi}
\end{array}\right) \\
\textbf{fi}
\end{array}\right)
\end{array}
$$

$EnterState \mathrel{\widehat=} s, tpp : State;\ ce : EVENT \bullet EnterState16(s, tpp, ce)$

$EnterState24 \mathrel{\widehat=} s, tpp : State;\ ce : EVENT \bullet EnterState2(s, tpp, ce);$
$\quad \mathbf{var}\ b : \mathbb{B} \bullet enterState2Check(s.identifier, b);$
$\quad \left(\begin{array}{l} \mathbf{if}\ b = \mathbf{True} \longrightarrow \mathbf{Skip} \\ [\!]\ b = \mathbf{False} \longrightarrow EnterState34(s, tpp, ce) \\ \mathbf{fi} \end{array} \right)$

$EnterState14 \mathrel{\widehat=} s, tpp : State;\ ce : EVENT \bullet EnterState1(s, tpp, ce);$
$\quad \mathbf{var}\ b : \mathbb{B} \bullet enterState1Check(s.identifier, b);$
$\quad \left(\begin{array}{l} \mathbf{if}\ b = \mathbf{True} \longrightarrow \mathbf{Skip} \\ [\!]\ b = \mathbf{False} \longrightarrow EnterState24(s, tpp, ce) \\ \mathbf{fi} \end{array} \right)$

$EnterState35 \mathrel{\widehat=} s, tpp : State;\ ce : EVENT \bullet EnterState34(s, tpp, ce);$
$\quad status!(s.identifier)?active \longrightarrow \left(\begin{array}{l} \mathbf{if}\ active = \mathbf{True} \longrightarrow EnterState5(s, tpp, ce) \\ [\!]\ active = \mathbf{False} \longrightarrow \mathbf{Skip} \\ \mathbf{fi} \end{array} \right)$

$EnterState25 \mathrel{\widehat=} s, tpp : State;\ ce : EVENT \bullet EnterState2(s, tpp, ce);$
$\quad \mathbf{var}\ b : \mathbb{B} \bullet enterState2Check(s.identifier, b);$
$\quad \left(\begin{array}{l} \mathbf{if}\ b = \mathbf{True} \longrightarrow \mathbf{Skip} \\ [\!]\ b = \mathbf{False} \longrightarrow EnterState35(s, tpp, ce) \\ \mathbf{fi} \end{array} \right)$

$EnterState15 \mathrel{\widehat=} s, tpp : State;\ ce : EVENT \bullet EnterState1(s, tpp, ce);$
$\quad \mathbf{var}\ b : \mathbb{B} \bullet enterState1Check(s.identifier, b);$
$\quad \left(\begin{array}{l} \mathbf{if}\ b = \mathbf{True} \longrightarrow \mathbf{Skip} \\ [\!]\ b = \mathbf{False} \longrightarrow EnterState25(s, tpp, ce) \\ \mathbf{fi} \end{array} \right)$

$EnterState56 \mathrel{\widehat=} s, tpp : State;\ ce : EVENT \bullet EnterState5(s, tpp, ce);$
$\quad status!(s.identifier)?active \longrightarrow \left(\begin{array}{l} \mathbf{if}\ active = \mathbf{True} \longrightarrow EnterState6(s, tpp, ce) \\ [\!]\ active = \mathbf{False} \longrightarrow \mathbf{Skip} \\ \mathbf{fi} \end{array} \right)$

$EnterState36 \mathrel{\widehat=} s, tpp : State;\ ce : EVENT \bullet EnterState34(s, tpp, ce);$
$\quad status!(s.identifier)?active \longrightarrow \left(\begin{array}{l} \mathbf{if}\ active = \mathbf{True} \longrightarrow EnterState56(s, tpp, ce) \\ [\!]\ active = \mathbf{False} \longrightarrow \mathbf{Skip} \\ \mathbf{fi} \end{array} \right)$

$EnterState26 \mathrel{\widehat=} s, tpp : State;\ ce : EVENT \bullet EnterState2(s, tpp, ce);$
$\quad \mathbf{var}\ b : \mathbb{B} \bullet enterState2Check(s.identifier, b);$
$\quad \left(\begin{array}{l} \mathbf{if}\ b = \mathbf{True} \longrightarrow \mathbf{Skip} \\ [\!]\ b = \mathbf{False} \longrightarrow EnterState36(s, tpp, ce) \\ \mathbf{fi} \end{array} \right)$

$EnterState16 \mathrel{\widehat=} s, tpp : State;\ ce : EVENT \bullet EnterState1(s, tpp, ce);$
$\quad \mathbf{var}\ b : \mathbb{B} \bullet enterState1Check(s.identifier, b);$
$\quad \left(\begin{array}{l} \mathbf{if}\ b = \mathbf{True} \longrightarrow \mathbf{Skip} \\ [\!]\ b = \mathbf{False} \longrightarrow EnterState26(s, tpp, ce) \\ \mathbf{fi} \end{array} \right)$

$$EnterState1 \;\widehat{=}\; s, tpp : State;\; ce : EVENT \bullet status!(s.parent)?active \longrightarrow$$
$$\left(\begin{array}{l} \textbf{if } active = \textbf{False} \longrightarrow \\ \quad\quad state!(s.parent)?p \longrightarrow EnterState14(p, tpp, ce) \\ [\!] \; active = \textbf{True} \longrightarrow \textbf{Skip} \\ \textbf{fi} \end{array}\right)$$

$$EnterState2 \;\widehat{=}\; s, tpp : State;\; ce : EVENT \bullet$$
$$\left(\begin{array}{l} \textbf{if } s.type = AND \longrightarrow \\ \left(\begin{array}{l} \textbf{if } s.left \neq nullstate.identifier \longrightarrow status!(s.left)?active \longrightarrow \\ \left(\begin{array}{l} \textbf{if } active = \textbf{True} \longrightarrow \textbf{Skip} \\ [\!] \; active \neq \textbf{True} \longrightarrow state!(s.left)?ls \longrightarrow EnterState15(ls, tpp, ce) \\ \textbf{fi} \end{array}\right) \\ [\!] \; s.left = nullstate.identifier \longrightarrow \textbf{Skip} \\ \textbf{fi} \end{array}\right) \\ [\!] \; s.type \neq AND \longrightarrow \textbf{Skip} \\ \textbf{fi} \end{array}\right)$$

$$EnterState3 \;\widehat{=}\; s, tpp : State;\; ce : EVENT \bullet activate!(s.identifier) \longrightarrow \textbf{Skip}$$

$$EnterState4 \;\widehat{=}\; s, tpp : State;\; ce : EVENT \bullet$$
$$executeentryaction!(s.identifier) \longrightarrow LocalEventEntry(s.identifier)$$

$$EnterState34 \;\widehat{=}\; s, tpp : State;\; ce : EVENT \bullet status.(s.identifier)?active \longrightarrow$$
$$\left(\begin{array}{l} \textbf{if } active = \textbf{True} \longrightarrow \textbf{Skip} \\ [\!] \; active = \textbf{False} \longrightarrow EnterState3(s, tpp, ce) \; ; \; EnterState4(s, tpp, ce) \\ \textbf{fi} \end{array}\right)$$

$$EnterState5 \;\widehat{=}\; s, tpp : State;\; ce : EVENT \bullet$$
$$EnterState5a(s, tpp, ce) \; ; \; EnterState5b(s, tpp, ce)$$

$$EnterState5a \;\widehat{=}\; s, tpp : State;\; ce : EVENT \bullet$$
$$\left(\begin{array}{l} \textbf{if } s.history = \textbf{True} \longrightarrow history!(s.identifier)?lsid \longrightarrow \\ \quad\quad \left(\begin{array}{l} \textbf{if } lsid \neq nullstate.identifier \longrightarrow state!lsid?ls \longrightarrow EnterState15(ls, tpp, ce) \\ [\!] \; lsid = nullstate.identifier \longrightarrow ExecuteDefaultTransition(s, tpp, ce) \\ \textbf{fi} \end{array}\right) \\ [\!] \; s.history = \textbf{False} \longrightarrow \\ \quad\quad \left(\begin{array}{l} \textbf{if } s.default \neq nulltransition.identifier \longrightarrow ExecuteDefaultTransition(s, tpp, ce) \\ [\!] s.default = nulltransition.identifier \longrightarrow \textbf{Skip} \\ \textbf{fi} \end{array}\right) \\ \textbf{fi} \end{array}\right)$$

$$EnterState5b \;\widehat{=}\; s, tpp : State;\; ce : EVENT \bullet$$
$$\left(\begin{array}{l} \textbf{if } s.decomposition = SET \wedge s.default = nulltransition.identifier \longrightarrow \\ \quad\quad EnterStates15(s.substates, tpp, ce) \\ [\!] \; s.decomposition \neq SET \vee s.default \neq nulltransition.identifier \longrightarrow \textbf{Skip} \\ \textbf{fi} \end{array}\right)$$

$$
\begin{array}{l}
EnterStates15 \mathrel{\widehat=} ss : \mathrm{seq}\, SID;\ tpp : State;\ ce : EVENT \bullet \\
\left(\begin{array}{l}
\textbf{if } \#ss = 0 \longrightarrow \textbf{Skip} \\
[]\ \#ss > 0 \longrightarrow status!(head\ ss)?active \longrightarrow \\
\left(\begin{array}{l}
\textbf{if } active = \textbf{False} \longrightarrow state!(head\ ss)?first \longrightarrow EnterState15(first, tpp, ce) \\
[]\ active = \textbf{True} \longrightarrow \textbf{Skip} \\
\textbf{fi}
\end{array}\right); \\
\textbf{var } b : \mathbb{B} \bullet \left(\begin{array}{l}
enterState15Check(head\ ss, b); \\
\left(\begin{array}{l}
\textbf{if } b = \textbf{True} \longrightarrow \textbf{Skip} \\
[]\ b = \textbf{False} \longrightarrow EnterStates15(tail\ ss, tpp, ce) \\
\textbf{fi}
\end{array}\right)
\end{array}\right) \\
\textbf{fi}
\end{array}\right)
\end{array}
$$

$$
\begin{array}{l}
EnterState6 \mathrel{\widehat=} s, tpp : State;\ ce : EVENT \bullet \\
\left(\begin{array}{l}
\textbf{if } s.type = AND \land s.right \neq nullstate.identifier \longrightarrow \\
\quad state!(s.right)?rs \longrightarrow EnterState(rs, tpp, ce) \\
[]\ s.type \neq AND \lor s.right = nullstate.identifier \longrightarrow EnterState7(s, tpp, ce) \\
\textbf{fi}
\end{array}\right)
\end{array}
$$

$$
\begin{array}{l}
EnterState7 \mathrel{\widehat=} s, tpp : State;\ ce : EVENT \bullet \\
\left(\begin{array}{l}
\textbf{if } s.type \neq CHART \longrightarrow \\
\quad state!(s.parent)?p \longrightarrow \left(\begin{array}{l}
\textbf{if } tpp \neq p \longrightarrow EnterState6(p, tpp, ce) \\
[]\ tpp = p \longrightarrow \textbf{Skip} \\
\textbf{fi}
\end{array}\right) \\
[]\ s.type = CHART \longrightarrow \textbf{Skip} \\
\textbf{fi}
\end{array}\right)
\end{array}
$$

$$
\begin{array}{l}
ExecuteState \mathrel{\widehat=} s : State;\ ce : EVENT \bullet status!(s.identifier)?active \longrightarrow \\
\left(\begin{array}{l}
\textbf{if } active = \textbf{True} \longrightarrow \\
\left(\begin{array}{l}
\textbf{var } success : \mathbb{B} \bullet \\
ExecuteTransition(s.outer, \langle\rangle, s, ce, success); \\
\left(\begin{array}{l}
\textbf{if } success = \textbf{True} \longrightarrow \textbf{Skip} \\
[]\ success = \textbf{False} \longrightarrow \\
\quad \left(\begin{array}{l}
executeduringaction!(s.identifier)!ce \longrightarrow \\
LocalEventDuring(s.identifier); \\
\textbf{var } b : \mathbb{B} \bullet \\
\left(\begin{array}{l}
duringActionCheck(s.identifier, b); \\
\left(\begin{array}{l}
\textbf{if } b = \textbf{True} \longrightarrow \textbf{Skip} \\
[]\ b = \textbf{False} \longrightarrow AlternativeExecution(s, ce) \\
\textbf{fi}
\end{array}\right)
\end{array}\right)
\end{array}\right) \\
\textbf{fi}
\end{array}\right)
\end{array}\right) \\
[]\, active = \textbf{False} \longrightarrow \textbf{Skip} \\
\textbf{fi}
\end{array}\right)
\end{array}
$$

$$
\begin{array}{l}
AlternativeExecution \mathrel{\widehat=} s : State;\ ce : EVENT \bullet \\
\left(\begin{array}{l}
\textbf{var } success : \mathbb{B} \bullet ExecuteTransition(s.inner, \langle\rangle, s, ce, success); \\
\left(\begin{array}{l}
\textbf{if } success = \textbf{True} \longrightarrow \textbf{Skip} \\
[]success = \textbf{False} \longrightarrow ExecuteSubstates(s, ce) \\
\textbf{fi}
\end{array}\right)
\end{array}\right)
\end{array}
$$

$$
\begin{array}{l}
ExecuteSubstates \mathrel{\widehat=} s : State;\ ce : EVENT \bullet \\
\left(\begin{array}{l}
\textbf{if } s.decomposition = SET \longrightarrow ExecuteParallelStates(s.substates, ce) \\
[]\ s.decomposition = CLUSTER \longrightarrow ExecuteSequentialStates(s.substates, ce) \\
\textbf{fi}
\end{array}\right)
\end{array}
$$

$$
\begin{array}{l}
ExecuteParallelStates \mathrel{\widehat{=}} ss : \mathrm{seq}\, SID;\ ce : EVENT \bullet \\
\left(\begin{array}{l}
\textbf{if } \#ss = 0 \longrightarrow \textbf{Skip} \\
[]\ \#ss > 0 \longrightarrow state!(head\ ss)?first \longrightarrow ExecuteState(first, ce); \\
\quad \textbf{var } b : \mathbb{B} \bullet \left(\begin{array}{l} executeStateCheck(head\ ss, b); \\ \left(\begin{array}{l} \textbf{if } b = \textbf{True} \longrightarrow \textbf{Skip} \\ []\ b = \textbf{False} \longrightarrow ExecuteParallelStates(tail\ ss, ce) \end{array}\right) \\ \textbf{fi} \end{array}\right) \\
\textbf{fi}
\end{array}\right)
\end{array}
$$

$$
\begin{array}{l}
ExecuteSequentialStates \mathrel{\widehat{=}} ss : \mathrm{seq}\, SID;\ ce : EVENT \bullet \\
\left(\begin{array}{l}
\textbf{if } \#ss = 0 \longrightarrow \textbf{Skip} \\
[]\ \#ss > 0 \longrightarrow status!(head\ ss)?active \longrightarrow \\
\left(\begin{array}{l} \textbf{if } active = \textbf{True} \longrightarrow state!(head\ ss)?first \longrightarrow ExecuteState(first, ce) \\ []\ active = \textbf{False} \longrightarrow ExecuteSequentialStates(tail\ ss, ce) \\ \textbf{fi} \end{array}\right) \\
\textbf{fi}
\end{array}\right)
\end{array}
$$

$$
\begin{array}{l}
ExitState \mathrel{\widehat{=}} s : State;\ ce : EVENT \bullet \\
\left(\begin{array}{l}
\left(\begin{array}{l}
\textbf{if } s.right \neq nullstate.identifier \longrightarrow status!(s.right)?active \longrightarrow \\
\left(\begin{array}{l} \textbf{if } active = \textbf{True} \longrightarrow state!(s.right)?rs \longrightarrow ExitState(rs, ce) \\ []\ active = \textbf{False} \longrightarrow \textbf{Skip} \\ \textbf{fi} \end{array}\right) \\
[]\ s.right = nullstate.identifier \longrightarrow \textbf{Skip} \\
\textbf{fi}
\end{array}\right); \\
ExitStates(s.substates, ce); \\
executeexitaction!(s.identifier) \longrightarrow LocalEventExit(s.identifier); \\
\quad \textbf{var } b : \mathbb{B} \bullet \left(\begin{array}{l} exitActionCheck(s.identifier, b); \\ \left(\begin{array}{l} \textbf{if } b = \textbf{True} \longrightarrow \textbf{Skip} \\ []\ b = \textbf{False} \longrightarrow deactivate!(s.identifier) \longrightarrow \textbf{Skip} \\ \textbf{fi} \end{array}\right) \end{array}\right)
\end{array}\right)
\end{array}
$$

$$
\begin{array}{l}
ExitStates \mathrel{\widehat{=}} ss : \mathrm{seq}\, SID;\ ce : EVENT \bullet \\
\left(\begin{array}{l}
\textbf{if } \#ss = 0 \longrightarrow \textbf{Skip} \\
[]\ \#ss > 0 \longrightarrow status!(last\ ss)?active \longrightarrow \\
\quad \left(\begin{array}{l} \textbf{if } active = \textbf{True} \longrightarrow state!(last\ ss)?l \longrightarrow ExitState(l, ce) \\ []\ active = \textbf{False} \longrightarrow \textbf{Skip} \\ \textbf{fi} \end{array}\right); \\
\quad \textbf{var } b : \mathbb{B} \bullet \left(\begin{array}{l} exitStateCheck(head\ ss, b); \\ \left(\begin{array}{l} \textbf{if } b = \textbf{True} \longrightarrow \textbf{Skip} \\ []\ b = \textbf{False} \longrightarrow ExitStates(front\ ss, ce) \\ \textbf{fi} \end{array}\right) \end{array}\right) \\
\textbf{fi}
\end{array}\right)
\end{array}
$$

$$
\begin{array}{l}
ExecuteChart \mathrel{\widehat{=}} ce : EVENT \bullet chart?c \longrightarrow status!(c.identifier)?active \longrightarrow \\
\left(\begin{array}{l}
\textbf{if } active = \textbf{True} \longrightarrow ExecuteActiveChart(c, ce) \\
[]\ active = \textbf{False} \longrightarrow ExecuteInactiveChart(c, ce) \\
\textbf{fi}
\end{array}\right)
\end{array}
$$

$$
\begin{array}{l}
ExecuteInactiveChart \mathrel{\widehat{=}} c : State;\ ce : EVENT \bullet activate!(c.identifier) \longrightarrow \\
\left(\begin{array}{l}
\textbf{if } c.default \neq nulltransition.identifier \vee c.decomposition = CLUSTER \longrightarrow \\
\quad ExecuteDefaultTransition(c, c, ce) \\
[\!] \ c.default = nulltransition.identifier \wedge c.decomposition = SET \longrightarrow \\
\quad \left(\begin{array}{l}
\textbf{if } c.substates = \langle\rangle \longrightarrow \textbf{Skip} \\
[\!] \ c.substates \neq \langle\rangle \longrightarrow \\
\quad state!(head(c.substates))?first \longrightarrow EnterState(first, c, ce) \\
\textbf{fi}
\end{array} \right) \\
\textbf{fi}
\end{array} \right)
\end{array}
$$

$$
\begin{array}{l}
ExecuteActiveChart \ \widehat{=} \ c : State;\ ce : EVENT \bullet \\
\left(\begin{array}{l}
\textbf{if } c.substates = \varnothing \longrightarrow ExecuteInactiveChart(c, ce) \\
[\!] \ c.substates \neq \varnothing \longrightarrow ExecuteSubstates(c, ce) \\
\textbf{fi}
\end{array} \right)
\end{array}
$$

$$
\begin{array}{l}
LocalEventEntry \mathrel{\widehat{=}} sid : SID \bullet \mu X \bullet \\
\left(\begin{array}{l}
local\_event?e?s \longrightarrow \left(\begin{array}{l}
TreatLocalEvent(e, s); \\
\textbf{var } b : \mathbb{B} \bullet \\
\left(\begin{array}{l}
entryActionCheck(sid, b); \\
\left(\begin{array}{l}
\textbf{if } b = \textbf{True} \longrightarrow \\
\quad interrupt.\textbf{True} \longrightarrow end\_action \longrightarrow \textbf{Skip} \\
[\!] \ b = \textbf{False} \longrightarrow interrupt.\textbf{False} \longrightarrow X \\
\textbf{fi}
\end{array} \right)
\end{array} \right)
\end{array} \right) \\
\Box \\
end\_action \longrightarrow \textbf{Skip}
\end{array} \right)
\end{array}
$$

$$
\begin{array}{l}
LocalEventDuring \mathrel{\widehat{=}} sid : SID \bullet \mu X \bullet \\
\left(\begin{array}{l}
local\_event?e?s \longrightarrow \left(\begin{array}{l}
TreatLocalEvent(e, s); \\
\textbf{var } b : \mathbb{B} \\
\bullet \left(\begin{array}{l}
duringActionCheck(sid, b); \\
\left(\begin{array}{l}
\textbf{if } b = \textbf{True} \longrightarrow \\
\quad interrupt.\textbf{True} \longrightarrow end\_action \longrightarrow \textbf{Skip} \\
[\!] \ b = \textbf{False} \longrightarrow interrupt.\textbf{False} \longrightarrow X \\
\textbf{fi}
\end{array} \right)
\end{array} \right)
\end{array} \right) \\
\Box \\
end\_action \longrightarrow \textbf{Skip}
\end{array} \right)
\end{array}
$$

$$
\begin{array}{l}
LocalEventExit \mathrel{\widehat{=}} sid : SID \bullet \mu X \bullet \\
\left(\begin{array}{l}
local\_event?e?s \longrightarrow \left(\begin{array}{l}
TreatLocalEvent(e, s); \\
\textbf{var } b : \mathbb{B} \\
\bullet \left(\begin{array}{l}
exitActionCheck(sid, b); \\
\left(\begin{array}{l}
\textbf{if } b = \textbf{True} \longrightarrow \\
\quad interrupt.\textbf{True} \longrightarrow end\_action \longrightarrow \textbf{Skip} \\
[\!] \ b = \textbf{False} \longrightarrow interrupt.\textbf{False} \longrightarrow X \\
\textbf{fi}
\end{array} \right)
\end{array} \right)
\end{array} \right) \\
\Box \\
end\_action \longrightarrow \textbf{Skip}
\end{array} \right)
\end{array}
$$

$$LocalEventCondition \mathrel{\widehat=} sid : SID \bullet \mu X \bullet \left(\begin{array}{l} local\_event?e?s \longrightarrow \left(\begin{array}{l} TreatLocalEvent(e, s); \\ \textbf{var } b : \mathbb{B} \\ \bullet \left(\begin{array}{l} conditionActionCheck(sid, b); \\ \left(\begin{array}{l} \textbf{if } b = \textbf{True} \longrightarrow \\ \quad interrupt.\textbf{True} \longrightarrow end\_action \longrightarrow \textbf{Skip} \\ [\!] \ b = \textbf{False} \longrightarrow interrupt.\textbf{False} \longrightarrow X \\ \textbf{fi} \end{array} \right) \end{array} \right) \end{array} \right) \\ \Box \\ end\_action \longrightarrow \textbf{Skip} \end{array} \right)$$

$$LocalEventTransition \mathrel{\widehat=} sid : SID \bullet \mu X \bullet \left(\begin{array}{l} local\_event?e?s \longrightarrow \left(\begin{array}{l} TreatLocalEvent(e, s); \\ \textbf{var } b : \mathbb{B} \\ \bullet \left(\begin{array}{l} transitionActionCheck(sid, b); \\ \left(\begin{array}{l} \textbf{if } b = \textbf{True} \longrightarrow \\ \quad interrupt.\textbf{True} \longrightarrow end\_action \longrightarrow \textbf{Skip} \\ [\!] \ b = \textbf{False} \longrightarrow interrupt.\textbf{False} \longrightarrow X \\ \textbf{fi} \end{array} \right) \end{array} \right) \end{array} \right) \\ \Box \\ end\_action \longrightarrow \textbf{Skip} \end{array} \right)$$

$$TreatLocalEvent \mathrel{\widehat=} e : EVENT;\ s : State \bullet \left(\begin{array}{l} \textbf{if } s.type = CHART \longrightarrow ExecuteChart(e) \\ [\!] \ s.type \neq CHART \longrightarrow ExecuteState(s, e) \\ \textbf{fi} \end{array} \right) ;\ end\_local\_execution \longrightarrow \textbf{Skip}$$

$$ExecuteEvent \mathrel{\widehat=} e : EVENT;\ v : \mathbb{B} \bullet \left(\begin{array}{l} \textbf{if } v = \textbf{True} \longrightarrow ExecuteChart(e) \\ [\!] \ v = \textbf{False} \longrightarrow \textbf{Skip} \\ \textbf{fi} \end{array} \right)$$

$$ExecuteEvents \mathrel{\widehat=} es : \operatorname{seq} EVENT;\ vs : \operatorname{seq} \mathbb{B} \bullet (\overset{\cdot}{;}\ \ i : id(1 \mathinner{..} \# es) \bullet ExecuteEvent(es(i), vs(i)))$$

$$\bullet\ (\mu X \bullet Step\ ;\ \ end\_cycle \longrightarrow X)$$

end