

$\langle Model \rangle ::= \langle ProcDSL \rangle \mid \langle SchDSL \rangle$

$\langle ProcDSL \rangle ::= \text{'def' 'process' '{' } [\langle ProcAttr \rangle] \langle Process \rangle^* \text{'}' } [\langle ProcConf \rangle] [\langle ProcInit \rangle]$

$\langle ProcAttr \rangle ::= \text{'attribute' '{' } \langle PAttr \rangle^* \text{'}' }$

$\langle PAttr \rangle ::= [\text{'var' | 'val'}] \langle Type \rangle \langle ID \rangle (\text{' , ' } \langle ID \rangle)^* [\text{' = ' } \langle Value \rangle] \text{' ; ' }$

$\langle Type \rangle ::= \text{'int' } \mid \text{'byte' } \mid \text{'clock' }$

$\langle Process \rangle ::= \text{'proctype' } \langle ID \rangle (\text{' (' } [\langle PramList \rangle] \text{') ' } \text{'{' } \langle AttAss \rangle^* \text{'}' }$

$\langle PramList \rangle ::= \langle PramAss \rangle (\text{' ; ' } \langle PramAss \rangle)^*$

$\langle PramAss \rangle ::= \langle Type \rangle \langle ID \rangle (\text{' , ' } \langle ID \rangle)^* \text{' = ' } \langle Value \rangle$

$\langle AttAss \rangle ::= [\text{'this' ' . '}] \langle ID \rangle \text{' = ' } (\langle Value \rangle \mid \langle ID \rangle) \text{' ; ' }$

$\langle ProcConf \rangle ::= \text{'config' '{' } \langle PConf \rangle^* \text{'}' }$

$\langle PConf \rangle ::= \langle SporadicP \rangle \mid \langle PeriodicP \rangle$

$\langle SporadicP \rangle ::= \text{'sporadic' 'process' } \langle Proc \rangle \text{'in' ' (' } \langle INT \rangle \text{' , ' } \langle INT \rangle \text{') ' } [\text{'limited' } \langle INT \rangle] \text{' ; ' }$

$\langle PeriodicP \rangle ::= \text{'periodic' 'process' } \langle Proc \rangle \text{'offset' ' = ' } \langle INT \rangle \text{'period' ' = ' } \langle INT \rangle [\text{'limited' } \langle INT \rangle] \text{' ; ' }$

$\langle Proc \rangle ::= \langle ID \rangle (\text{' (' } [\langle Value \rangle (\text{' , ' } \langle Value \rangle)^*] \text{') ' }$

$\langle ProcInit \rangle ::= \text{'init' '{' } [\text{' [' } \langle PSet \rangle (\text{' , ' } \langle PSet \rangle)^* \text{'] ' } \text{'}' } \text{' ; ' }$

$\langle PSet \rangle ::= \text{'{' } \langle Proc \rangle (\text{' , ' } \langle Proc \rangle)^* \text{'}' }$

Figure 1: The grammar for specifying the process attributes.

$\langle SchDSL \rangle ::= \langle SchDef \rangle [\langle OrdDef \rangle] [\langle Verify \rangle]$
 $\langle SchDef \rangle ::= \text{'scheduler' } \langle ID \rangle \text{'(' } [\langle ParamList \rangle] \text{'(' } [\text{'refines' } \langle ID \rangle] \text{'(' } [\langle Generate \rangle] [\langle VarDef \rangle] [\langle DatDef \rangle] [\langle HandlerDef \rangle] [\langle InterDef \rangle] \text{'}' }$
 $\langle VarDef \rangle ::= \text{'variable' '{' } \langle VDec \rangle^* \text{'}' }$
 $\langle VDec \rangle ::= [\langle IfDef \rangle] (\langle VBlockDef \rangle \mid \langle VOneDef \rangle)$
 $\langle IfDef \rangle ::= \text{'\#'} \text{'ifdef' '(' } \langle Expr \rangle \text{'(' }$
 $\langle VBlockDef \rangle ::= \text{'{' } \langle VOneDef \rangle^* \text{'}' }$
 $\langle VOneDef \rangle ::= \langle Type \rangle \langle ID \rangle \text{'(' } \langle ID \rangle^* [\text{'=' } \langle Value \rangle] \text{';' }$
 $\langle DatDef \rangle ::= \text{'data' '{' } \langle DDef \rangle^* \text{'}' }$
 $\langle DDef \rangle ::= [\langle IfDef \rangle] \text{'data' } (\langle DBlockDef \rangle \mid \langle DOneDef \rangle)$
 $\langle DBlockDef \rangle ::= \text{'{' } \langle DOneDef \rangle^* \text{'}' }$
 $\langle DOneDef \rangle ::= \langle VOneDef \rangle \mid \langle ColDef \rangle$
 $\langle ColDef \rangle ::= [\text{'refines' } \text{'collection' } \langle ID \rangle [\text{'using' } \langle ID \rangle \text{'(' } \langle ID \rangle^*] [\text{'with' } \langle OrdType \rangle] \text{';' }$
 $\langle OrdType \rangle ::= \text{'lifo' } \mid \text{'fifo' }$
 $\langle HandlerDef \rangle ::= \text{'event' 'handler' '{' } \langle EventDef \rangle^* \text{'}' }$
 $\langle EventDef \rangle ::= \langle Event \rangle \text{'(' } [\langle ID \rangle] \text{'(' } \text{'{' } \langle IfDefStm \rangle^* \text{'}' }$
 $\langle IfDefStm \rangle ::= [\langle IfDef \rangle] \langle Stm \rangle$
 $\langle Event \rangle ::= \text{'select_process' } \mid \text{'new_process' } \mid \text{'clock' } \mid \text{'pre_take' } \mid \text{'post_take' } \mid \text{'action' }$
 $\langle InterDef \rangle ::= \text{'interface' '{' } \langle InterFunc \rangle^* \text{'}' }$
 $\langle InterFunc \rangle ::= \text{'function' } \langle ID \rangle \text{'(' } [\langle IParamList \rangle] \text{'(' } \text{'{' } \langle Stm \rangle^* \text{'}' }$
 $\langle IParamList \rangle ::= \langle IParamDec \rangle \text{'(' } \langle IParamDec \rangle^*$
 $\langle IParamDec \rangle ::= \langle Type \rangle \langle ID \rangle$
 $\langle OrdDef \rangle ::= \text{'comparator' '{' } [\langle CVarDef \rangle] \langle CompDef \rangle^* \text{'}' }$
 $\langle CVarDef \rangle ::= \text{'variable' '{' } \langle VOneDef \rangle^* \text{'}' }$
 $\langle CompDef \rangle ::= \text{'comparetype' } \langle ID \rangle \text{'(' } \langle ID \rangle \text{'process' } \langle ID \rangle \text{'(' } \langle ID \rangle \text{'(' } \text{'{' } \langle Stm \rangle^* \text{'}' }$

Figure 2: The grammar for specifying the scheduling policy.

$\langle Stm \rangle ::= \langle SetTime \rangle \mid \langle SetCol \rangle \mid \langle Change \rangle \mid \langle Move \rangle \mid \langle Remove \rangle \mid \langle Get \rangle \mid \langle New \rangle$
 $\quad \mid \langle If \rangle \mid \langle Loop \rangle \mid \langle Block \rangle \mid \langle Assert \rangle \mid \langle Print \rangle \mid \langle Return \rangle \mid \langle Gen \rangle \mid \langle GenLn \rangle$

$\langle SetTime \rangle ::= \text{'time_slice' '=' } \langle Expr \rangle \text{' ;'}$

$\langle SetCol \rangle ::= \text{'return_set' '=' } \langle ID \rangle \text{' ;'}$

$\langle Change \rangle ::= \langle ChgUnOp \rangle \mid \langle ChgExpr \rangle$

$\langle ChgUnOp \rangle ::= \langle QualName \rangle \text{' ++' } \mid \text{' \{\} ' } \text{' ;'}$

$\langle ChgExpr \rangle ::= \langle QualName \rangle \text{' = ' } \langle Expr \rangle \text{' ;'}$

$\langle QualName \rangle ::= \langle ID \rangle \text{' [' . ' } \langle ID \rangle \text{']}$

$\langle Move \rangle ::= \text{'move' } \langle ID \rangle \text{ to } \langle ID \rangle \text{' ;'}$

$\langle Remove \rangle ::= \text{'remove' } \langle ID \rangle \text{' ;'}$

$\langle Get \rangle ::= \text{'get' 'process' 'from' } \langle ID \rangle \text{' to' 'run' ' ;'}$

$\langle New \rangle ::= \text{'new' } \langle Proc \rangle \text{' [, ' } \langle INT \rangle \text{'] ;'}$

$\langle If \rangle ::= \text{'if' ' (' } \langle Expr \rangle \text{')' } \langle Stm \rangle \text{' ['else' } \langle Stm \rangle \text{']}$

$\langle Loop \rangle ::= \text{'for' 'each' 'process' } \langle ID \rangle \text{' in' } \langle ID \rangle \text{' } \langle Stm \rangle$

$\langle Block \rangle ::= \text{'{' } \langle Stm \rangle^* \text{' \}'}$

$\langle Assert \rangle ::= \text{'assert' } \langle Expr \rangle \text{' ;'}$

$\langle Print \rangle ::= \text{'print' } \langle Expr \rangle \text{' ;'}$

$\langle Return \rangle ::= \text{'return' } \langle OrderType \rangle \text{' ;'}$

$\langle OrderType \rangle ::= \text{'greater' } \mid \text{'less' } \mid \text{'equal'}$

$\langle Gen \rangle ::= \text{'gen' [} \langle ID \rangle \text{' , ' } \langle Expr \rangle \text{' ;'}$

$\langle GenLn \rangle ::= \text{'genln' [} \langle ID \rangle \text{' , ' } \langle Expr \rangle \text{' ;'}$

Figure 3: The grammar for specifying the statements.

$\langle Expr \rangle ::= \langle Or \rangle$
 $\langle Or \rangle ::= \langle And \rangle ('||' \langle And \rangle)^*$
 $\langle And \rangle ::= \langle Equality \rangle ('\&\&' \langle Equality \rangle)^*$
 $\langle Equality \rangle ::= \langle Equality \rangle ('==' | '!=') \langle Compar \rangle$
 $\langle Compar \rangle ::= \langle PlusMinus \rangle ('>=' | '<=' | '>' | '<') \langle PlusMinus \rangle$
 $\langle PlusMinus \rangle ::= \langle MulOrDiv \rangle ('+' | '-') \langle MulOrDiv \rangle$
 $\langle MulOrDiv \rangle ::= \langle MulOrDiv \rangle ('*' | '/') \langle Primary \rangle$
 $\langle Primary \rangle ::= 'C' \langle Expr \rangle ' ' | '!' \langle Primary \rangle | \langle Empty \rangle | \langle Null \rangle | \langle InCol \rangle | \langle Exist \rangle$
 $\quad | \langle GetID \rangle | \langle HasName \rangle | \langle Atomic \rangle$
 $\langle Empty \rangle ::= \langle ID \rangle '.' 'isEmpty' '(' ' ' '$
 $\langle Null \rangle ::= \langle ID \rangle '.' 'isNull' '(' ' ' '$
 $\langle InCol \rangle ::= \langle ID \rangle '.' 'containsProcess' '(' \langle STRING \rangle ' ' '$
 $\langle Exist \rangle ::= 'exists' '(' \langle STRING \rangle ' ' '$
 $\langle GetID \rangle ::= 'get_pid' '(' \langle STRING \rangle ' ' '$
 $\langle HasName \rangle ::= \langle ID \rangle '.' 'hasName' '(' \langle STRING \rangle ' ' '$
 $\langle Atomic \rangle ::= \langle Value \rangle | \langle QualName \rangle | \langle SysVar \rangle$
 $\langle SysVar \rangle ::= 'Sys' '(' \langle ID \rangle ' ' '$

Figure 4: The grammar for specifying the expression.

$\langle Generate \rangle ::= \text{'generate' } \{ \langle GenConfig \rangle \langle GenComp \rangle \}$
 $\langle GenConfig \rangle ::= \text{'configuration' } \{ [\langle GenOption \rangle \text{' ;' }] [\langle Dir \rangle \text{' ;' }] [\langle FName \rangle \text{' ;' }] [\langle FExt \rangle \text{' ;' }] \text{'test' } (\text{'program' } | \text{'case' } | \text{'data'}) \text{'=' } \langle TestPart \rangle \}$
 $\langle GenOption \rangle ::= \text{'option' } \text{'=' } \{ \langle GenOpt \rangle (\text{' , ' } \langle GenOpt \rangle)^* \}$
 $\langle GenOpt \rangle ::= \text{'Searching' } | \text{'Error' } | \text{'Property' } | \text{'All'}$
 $\langle Dir \rangle ::= \text{'directory' } \text{'=' } \langle STRING \rangle \text{' ;'}$
 $\langle FName \rangle ::= \text{'file' } \text{'name' } \text{'=' } \langle STRING \rangle \text{' ;'}$
 $\langle FExt \rangle ::= \text{'file' } \text{'extension' } \text{'=' } \langle STRING \rangle \text{' ;'}$
 $\langle TestPart \rangle ::= \langle GenPart \rangle (\text{' + ' } \langle GenPart \rangle)^*$
 $\langle GenPart \rangle ::= \text{'(' } [\langle STRING \rangle \text{' + '}] (\langle ID \rangle | \text{'init' } | \text{'processes' } | \text{'behaviors' } | \text{'error'}) [\text{' + ' } \langle STRING \rangle]^* \text{')'}$
 $\langle GenComp \rangle ::= \text{'component' } '' (\langle Comp \rangle)^* [\langle InitGen \rangle] [\langle ProcGen \rangle] '';$
 $\langle Comp \rangle ::= \langle ID \rangle \{ \langle Gen \rangle | \langle GenLn \rangle \}^* \}$
 $\langle InitGen \rangle ::= \text{'init' } \{ \langle Template \rangle \}$
 $\langle ProcGen \rangle ::= \text{'process' } \{ \langle Template \rangle \}$
 $\langle Template \rangle ::= [\langle SetTemplate \rangle] \langle Behavior \rangle$
 $\langle SetTemplate \rangle ::= \text{'template' } \text{'=' } \langle Expr \rangle \text{' ;'}$
 $\langle Behavior \rangle ::= \text{'behavior' } \text{'=' } \langle EventTemp \rangle (\text{' + ' } \langle EventTemp \rangle)^* \text{' ;'}$
 $\langle EventTemp \rangle ::= \text{'(' } [\langle Expr \rangle \text{' + '}] \langle Event \rangle [\text{' + ' } \langle Expr \rangle]^* \text{')'}$

Figure 5: The grammar for specifying the test generation.

$$\begin{aligned}
\langle \textit{Verify} \rangle &::= \text{'verify' } \{ \langle \textit{CTL_AT} \rangle \} \langle \textit{CTL} \rangle \text{'}' \\
\langle \textit{CTL_AT} \rangle &::= \text{'@' } \langle \textit{Expr} \rangle \text{'.'} \\
\langle \textit{CTL} \rangle &::= \text{'(' } \langle \textit{Expr} \rangle \text{'') } \mid \text{'not' } \langle \textit{CTL} \rangle \mid \text{'or' } \langle \textit{CTL} \rangle \langle \textit{CTL} \rangle \mid \text{'implies' } \langle \textit{CTL} \rangle \\
&\quad \langle \textit{CTL} \rangle \mid \text{'AX' } \langle \textit{CTL} \rangle \mid \text{'AF' } \langle \textit{CTL} \rangle \mid \text{'AG' } \langle \textit{CTL} \rangle \mid \text{'EX' } \langle \textit{CTL} \rangle \mid \text{'EF' } \langle \textit{CTL} \rangle \mid \\
&\quad \text{'EG' } \langle \textit{CTL} \rangle \mid \text{'AU' } \langle \textit{CTL} \rangle \langle \textit{CTL} \rangle \mid \text{'EU' } \langle \textit{CTL} \rangle \langle \textit{CTL} \rangle
\end{aligned}$$

Figure 6: The grammar for specifying the property.