Grammar of symsim

Bernard van Gastel

Sunday 15th March, 2015

Abstract

Syntax as supported by the xMAS tools.

We support the following syntax for matching expressions, as described by this BNF-grammar:

⟨*source-expr*⟩ ::= ⟨*expr*⟩
| ⟨*expr*⟩ '`->`' ⟨*spec*⟩

⟨*spec*⟩ ::= '`(`' ⟨*node*⟩ '`,`' ⟨*expr*⟩ '`)`'
| ⟨*spec*⟩ ⟨*logical-op*⟩ ⟨*spec*⟩

⟨*expr*⟩ ::= ⟨*enum-match*⟩ | ⟨*integer-match*⟩
| '`(`' ⟨*expr*⟩ '`)`'
| '`!`' ⟨*expr*⟩
| ⟨*expr*⟩ '`?`' ⟨*expr*⟩ '`:`' ⟨*expr*⟩
| ⟨*expr*⟩ ⟨*logical-op*⟩ ⟨*expr*⟩

⟨*logical-op*⟩ ::= '`and`' | '`&&`' | '`or`' | '`||`'

⟨*enum-match*⟩ ::= ⟨*variable*⟩
| ⟨*variable*⟩ '`in`' '`{`' ⟨*enum-contents*⟩ '`}`'
| ⟨*variable*⟩ '`not`' '`in`' '`{`' ⟨*enum-contents*⟩ '`}`'

⟨*enum-contents*⟩ ::= ⟨*label*⟩ | ⟨*label*⟩ '`,`' ⟨*enum-contents*⟩

⟨*interval*⟩ ::= ⟨*constant*⟩
| '`[`' ⟨*constant*⟩ '`..`' ⟨*constant*⟩ '`]`'

⟨*integer-match*⟩ ::= ⟨*variable*⟩
| ⟨*variable*⟩ ⟨*compare-op*⟩ ⟨*constant*⟩

| ⟨*variable*⟩ '`in`' ⟨*interval*⟩
| ⟨*variable*⟩ '`not`' '`in`' ⟨*interval*⟩

⟨*compare-op*⟩ ::= '`<`' | '`<=`' | '`>=`' | '`>`' | '`==`'

⟨*constant*⟩ ::= ⟨*integer*⟩
| ⟨*constant*⟩ ⟨*constant-op*⟩ ⟨*constant*⟩
| '`(`' ⟨*constant*⟩ '`)`'

⟨*constant-op*⟩ ::= '`+`' | '`-`' | '`*`' | '`/`' | '`%`' | '`^`'

For definition of modifying expressions we support another syntax, as they express another kind of intent. The syntax for these expressions is described by the following BNF-grammar:

⟨*expr*⟩ ::= ⟨*assignment*⟩
| '`if`' ⟨*expr-from-matching-expressions*⟩ '`then`' ⟨*assignment*⟩
'`else`' ⟨*expr*⟩

⟨*assignment*⟩ ::= ⟨*field-definition*⟩
| ⟨*field-definition*⟩ '`,`' ⟨*assignment*⟩

⟨*field-definition*⟩ ::= ⟨*variable*⟩ '`:=`' ⟨*value-expr*⟩

⟨*value-expr*⟩ ::= ⟨*variable*⟩ | ⟨*integer-expr*⟩
| ⟨*variable*⟩ '`with`' '`{`' ⟨*substitution-def*⟩ '`}`'

⟨*integer-expr*⟩ ::= ⟨*constant*⟩ | ⟨*variable*⟩ | '`(`' ⟨*integer-expr*⟩ '`)`'
| ⟨*integer-expr*⟩ ⟨*arithmetic-op*⟩ ⟨*integer-expr*⟩
| '`[`' ⟨*constant*⟩ '`..`' ⟨*constant*⟩ '`]`'

⟨*arithmetic-op*⟩ ::= '`+`' | '`-`' | '`*`' | '`/`'

⟨*substitution-def*⟩ ::= ⟨*label*⟩ '`:`' ⟨*label*⟩
| ⟨*label*⟩ '`:`' ⟨*label*⟩ '`,`' ⟨*substitution-def*⟩

If a substitution is defined on an expression, a special label '`_`' can be defined, which is the default (fail-over) case of the substitution.