# SynDEx v7 Grammar

**Julien Forget, Maxence Guesdon, Cécile Stentzel**

27 août 2009

# Table des matières

**1 Conventions** **2**

**2 Keywords and base types** **3**

2.1 General application information . . . . . 3
2.2 Operation Groups were previously called Software Components . . . . . 3
2.3 Algorithm . . . . . 3
2.4 Architecture . . . . . 4
2.5 Adequation result . . . . . 4
2.6 Misc . . . . . 5
2.7 Symbols . . . . . 5

**3 Regular expressions** **6**

**4 Application specification** **7**

# Chapitre 1

# Conventions

- upper-case text stands for keywords or base types ;
- lower-case text stands for rule names ;
- the first square brackets [...] for a rule only delimit the description of the rule ;
- the second square brackets (inside a rule description) [...] represent optional elements ;
- curly brackets {...} represent zero, one or several repetitions of the enclosed element ;
- pipes | represent alternatives for a rule ;
- usual brackets (...) are used inside a rule for sub-alternatives ;

# Chapitre 2

# Keywords and base types

## 2.1 General application information

```
"include":                INCLUDE
"def":                    DEF
"main":                   MAIN
"application":            APPLICATION
"description":            DESCRIPTION
```

## 2.2 Operation Groups were previously called Software Components

```
"operation_group":        OG
"software_component":     XSC
"constraint":             CONSTRAINT
"absolute":               ABSOLUTE
"relative":               RELATIVE
"union":                  UNION
"disjunction":            DISJUNCTION
"syndex_version":         SYNDEX_VERSION
"initseq":                INIT_SEQ
"loopseq":                LOOP_SEQ
"endseq":                 END_SEQ
"code_phases":            CODE_PHASES
```

## 2.3 Algorithm

```
"constant":               CONSTANT
"sensor":                 SENSOR
"actuator":               ACTUATOR
"memory":                 DELAY
"algorithm":              ALGORITHM
"internal":               INTERNAL
"attach_all":             ATTACH_ALL
"attach_ref":             ATTACH_REF
```

```
"attach_condi":             ATTACH_CONDI
"attach_condo":             ATTACH_CONDO
"attach_explode":           ATTACH_EXPLODE
"attach_implode":           ATTACH_IMPLODE
"conditions":               CONDITIONS
"references":               REFERENCES
"dependences":              DEPENDANCES
"strong_precedence_data":   STRONGPRECEDENCEDATA
"weak_precedence_data":     WEAKPRECEDENCEDATA
"precedence":               PRECEDENCE
"data":                     DATA
"condition_synchro":        CONDITION_SYNCHRO
```

## 2.4 Architecture

```
"architecture":             ARCHITECTURE
"operator":                 OPERATOR
"operators":                OPERATORS
"gate":                     GATE
"media":                    MEDIA
"medias":                   MEDIAS
"sampp":                    SAMPP
"sammp":                    SAMMP
"ram":                      RAM
"broadcast":                BROADCAST
"no_broadcast":             NOBROADCAST
"extra_durations_operator": EXTRA_DURATIONS_OPERATOR
"extra_durations_media":    EXTRA_DURATIONS_MEDIA
"connections":              CONNECTIONS
```

## 2.5 Adequation result

```
"ports":                    PORTS
"schedules":                SCHEDULES
"operation_scheduled":      OPERATION_SCHEDULED
"scheduled":                SCHEDULED
"calcul":                   CALCUL
"communication":            COMMUNICATION
"send":                     SEND
"receive":                  RECEIVE
"sync":                     SYNC
"send_synchro":             SEND_SYNCHRO
"receive_synchro":          RECEIVE_SYNCHRO
"read":                     READ
"write":                    WRITE
"ihm":                      IHM
"condI":                    CONDI
"condO":                    CONDO
"explode":                  EXPLODE
"implode":                  IMPLODE
```

```
"synchro_constant":         SYNCHRO_CONSTANT
"cond_level":               COND_LEVEL
"schedule_dependences":     SCHEDULE_DEPENDENCES
"schedule_conditions":      SCHEDULE_CONDITIONS
```

## 2.6 Misc

```
"on":                       ON
"true":                     TRUE
"false":                   FALSE
```

## 2.7 Symbols

```
   eof:           EOF
   "?":           IN
   "!":           OUT
   "->":          TO
   "<-":          BACKARROW
   ’@’:           AT
   ’=’:           EQU
   ’\\’:          BACKSLASH
   ’/’:           DIV
   ’-’:           MINUS
   ’|’:           BAR
   ’[’:           LDIM
   ’]’:           RDIM
   ’<’:           LARG
   ’>’:           RARG
   ’{’:           LLIST
   ’}’:           RLIST
   ’(’:           LPAR
   ’)’:           RPAR
   ’&’:           AND
   ’:’:           COL
’#’[^\n]*:        COMMENT
’"’[^’"’]*’"’*:   STRING
[’a’-’z’’A’-’Z’’_’][’a’-’z’’A’-’Z’’_’’-’’0’-’9’’*’]*: NAME ( if not a keyword )
[’+’’-’]?[’0’-’9’]+(’.’[’0’-’9’]*)?(’e’[’+’’-’]?[’0’-’9’]+)?: FLOAT
```

# Chapitre 3

# Regular expressions

expr_list ::= [ [ expr_list_continue ] ]
expr_list_continue ::= [ { expr COMMA } expr ]
expr ::= [ NAME | FLOAT | STRING | LPAREN expr RPAREN | expr PLUS expr | expr MINUS expr | expr TIMES expr | expr DIV expr | CEIL expr | MINUS expr | LLIST expr_list RLIST | BAR expr BAR ]
expression ::= expr EOE

# Chapitre 4

# Application specification

The entry point of the application is the *file* rule.