

SynDEX v7 Grammar

Julien Forget, Maxence Guesdon, Cécile Stentzel

23 septembre 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
"sampp":	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.

def_desc	::=	[[DESCRIPTION COL STRING]]
rfc_desc	::=	[[STRING]]
comment	::=	COMMENT
boolean	::=	[FALSE TRUE]
int	::=	FLOAT
integer	::=	[[MINUS] int]
name_list	::=	[{ NAME }]
rfc	::=	[NAME [DIV NAME]]
rfc_name	::=	rfc DOT NAME
rfc_name_list	::=	[{ rfc_name }]
rfc_path	::=	[{ BACKSLASH [NAME] }]
attachement_type	::=	[ATTACH_ALL ATTACH_REF ATTACH_CONDI ATTACH_CONDO ATTACH_EXPLODE ATTACH_IMPLODE]
operation_attached	::=	[LDIM rfc_path (RDIM COMMA attachement_type RDIM)]
operation_attached_list	::=	[{ operation_attached }]
expression	::=	
arg_names_list	::=	[{ NAME SCOL } NAME]
arg_names	::=	[[LARG arg_names_list RARG]]
arg_vals_list	::=	[{ expression SCOL } expression]
arg_vals	::=	[[LARG arg_vals_list RARG]]
dimension	::=	[[LDIM expression RDIM]]
range	::=	[[LDIM expression (RDIM DOT DOT expression RDIM)]]
coord2d	::=	integer COMMA integer
period_port	::=	[[int]]
rfc_prd	::=	[[int]]
rank	::=	[[int]]
pos	::=	[[AT coord2d]]
dim_window	::=	[[coord2d]]
version	::=	SYNDEX_VERSION COL STRING
code_phase	::=	[INIT_SEQ LOOP_SEQ END_SEQ]
code_phase_list	::=	[{ code_phase }]
code_phases	::=	[[CODE_PHASES COL code_phase_list SCOL]]

in_port	::=	IN NAME dimension NAME rank pos SCOL period_port
out_port	::=	OUT NAME dimension NAME rank pos SCOL period_port
inout_port	::=	AND NAME dimension NAME rank pos SCOL period_port
in_port_list	::=	[{ in_port }]
out_port_list	::=	[{ out_port }]
port_list	::=	[{ (in_port out_port inout_port) }]
dpd_prt	::=	[NAME [DOT NAME]]
abstract	::=	[[TRUE FALSE]]
dpd_rfc	::=	NAME
dependence	::=	[(STRONGPRECEDENCEDATA dpd_prt TO dpd_prt WEAKPRECEDENCEDATA dpd_prt TO dpd_prt PRECEDENCE dpd_rfc TO dpd_rfc DATA dpd_prt TO dpd_prt) SCOL]
dependence_list	::=	[{ dependence }]
rep_prts	::=	[NAME BACKARROW NAME [COMMA rep_prts]]
rep	::=	[[LDIM expression (RDIM COL rep_prts RDIM)]]
reference	::=	rfc arg_vals rep NAME pos rfc_desc rfc_prd abstract SCOL
reference_list	::=	[{ reference }]
condition_algo	::=	[[boolean NAME EQU integer]]
condition	::=	CONDITIONS COL condition_algo SCOL
references	::=	REFERENCES COL reference_list
dependences	::=	DEPENDANCES COL dependence_list
cnd_rfcs_dpds	::=	condition references dependences
cnd_rfcs_dpds_list	::=	[{ cnd_rfcs_dpds }]
internal	::=	DEF INTERNAL NAME arg_names COL port_list
constant	::=	DEF CONSTANT NAME arg_names dim_window COL out_port_list def_desc
sensor	::=	DEF SENSOR NAME arg_names dim_window COL out_port_list def_desc
actuator	::=	DEF ACTUATOR NAME arg_names dim_window COL in_port_list def_desc
delay	::=	DEF DELAY NAME range arg_names dim_window COL port_list def_desc
algorithm	::=	DEF ALGORITHM NAME arg_names dim_window COL port_list cnd_rfcs_dpds_list code_phases def_desc
algo	::=	[internal constant sensor actuator delay algorithm]
bus_type	::=	[(SAMPP SAMMP RAM) SCOL]
broadcast	::=	[[BROADCAST NOBROADCAST]]
gate	::=	GATE NAME NAME SCOL
gate_list	::=	[{ gate }]
duration	::=	rfc EQU FLOAT SCOL
durations_list	::=	[{ duration }]
gateref	::=	NAME DOT NAME
operatorref	::=	rfc NAME pos SCOL
operatorref_list	::=	[{ operatorref }]
mediaref	::=	rfc NAME broadcast pos SCOL
mediaref_list	::=	[{ mediaref }]

connection	::=	gateref NAME SCOL
connection_list	::=	[{ connection }]
operators	::=	OPERATORS COL operatorref_list
medias	::=	MEDIAS COL mediaref_list
connections	::=	CONNECTIONS COL connection_list
main_operator	::=	[[MAIN OPERATOR NAME SCOL]]
operator	::=	DEF OPERATOR NAME COL gate_list durations_list def_desc code phases
media	::=	DEF MEDIA NAME COL bus_type durations_list def_desc
extra_durations_operator	::=	EXTRA_DURATIONS_OPERATOR rfc COL durations_list
extra_durations_media	::=	EXTRA_DURATIONS_MEDIA rfc COL durations_list
architecture	::=	DEF ARCHITECTURE NAME dim_window COL operators main_operator medias connections def_desc
archi	::=	[operator media extra_durations_operator extra_durations_media architecture]
main	::=	[MAIN (ALGORITHM rfc arg_vals SCOL ARCHITECTURE rfc SCOL)]
xsc_definition	::=	[(XSC OG) NAME COL operation_attached_list SCOL]
operationonproc	::=	CONSTRAINT COL rfc_path ON rfc_name_list SCOL
absoluteconstraint	::=	ABSOLUTE CONSTRAINT COL NAME ON rfc_name_list SCOL
relativeconstraint_type	::=	[UNION DISJUNCTION]
relativeconstraint	::=	RELATIVE CONSTRAINT COL relativeconstraint_type name_list SCOL
constraints	::=	[operationonproc absoluteconstraint relativeconstraint]
description	::=	APPLICATION def_desc
calcul_path	::=	[[DIV calcul_path] DIV NAME]
communication_name	::=	[{ NAME COMMA } NAME]
string_path	::=	[[DIV string_path] DIV NAME]
communication_path_not_repeated	::=	[DIV communication_name LPAR string_path DOT NAME RPAR]
communication_path	::=	[communication_path_not_repeated [NAME]]
operation_path	::=	[calcul_path communication_path]
operator_list	::=	[{ NAME COMMA } NAME]
receivers	::=	LPAR operator_list RPAR
operation_port	::=	operation_path DOT NAME
calcul_class	::=	[CONSTANT SENSOR ACTUATOR DELAY ALGORITHM INTERNAL]
communication_class	::=	[WRITE NAME operation_port READ NAME operation_port SEND NAME receivers operation_port RECEIVE NAME receivers NAME operation_port SYNC NAME receivers NAME operation_port SEND_SYNCHRO NAME NAME RECEIVE_SYNCHRO NAME NAME]
opn_class	::=	[CALCUL calcul_class rfc COMMUNICATION communication_class]
origin	::=	[(IHM CONDI CONDO EXPLODE IMplode SYNCHRO_CONSTANT) operation_path]
opn_title	::=	opn_class arg_vals LPAR origin RPAR SCOL

operator_class	::=	[(OPERATOR MEDIA) NAME]
schedule_place	::=	SCHEDULED COL operator_class integer FLOAT integer
adeq_condition	::=	operation_port EQU integer
adeq_cond_list	::=	[{ adeq_condition AND } adeq_condition]
adeq_conditions	::=	[CONDITIONS COL (boolean adeq_cond_list)]
dir	::=	[IN OUT AND]
port_class	::=	[DATA PRECEDENCE DELAY CONDITION_SYNCHRO]
adeq_port	::=	dir NAME LDIM int RDIM NAME port_class integer period_port SCOL
adeq_port_list	::=	[{ adeq_port }]
adeq_ports	::=	PORTS COL adeq_port_list
adeq_dependence	::=	[(STRONGPRECEDENCEDATA COND_LEVEL EQU int operation_port TO operation_port STRONGPRECEDENCEDATA operation_port TO operation_port PRECEDENCE operation_path TO operation_path) adeq_conditions SCOL]
adeq_dpd_list	::=	[{ adeq_dependence }]
adeq_dpds	::=	SCHEDULE_DEPENDENCES COL adeq_dpd_list
operation_condition	::=	operation_path adeq_conditions
operation_condition_list	::=	[{ operation_condition }]
adeq_operations_conditions	::=	SCHEDULE_CONDITIONS operation_condition_list
operation_scheduled	::=	OPERATION_SCHEDULED operation_path COL opn_title schedule_place adeq_ports
schedules	::=	SCHEDULES COL
command	::=	[version description algo archi main schedules operation_scheduled adeq_dpds adeq_operations_conditions xsc_definition constraints comment]
command_list	::=	[{ command }]
fileinclude	::=	INCLUDE STRING SCOL
file	::=	[command_list (fileinclude EOF)]