| BPELVT<br>BPEL Verification TOOL |  <br>ReDCAD Laboratory, University of Sfax , Tunisia |
|---|---|

# BPEL Grammar

We present here all the grammar rules of BPEL4WS language.

system: proc+;

proc : BEGIN_PROCSS name targetnamsp supjoinFail (xmlns)+ OPE (importact)+ partnerlinks vars (msgexchangs)? (correlationsset)? (actbpel)+ END_PROCSS;

targetnamsp : 'targetNamespace=' Ponct ID Ponct ID Ponct;

supjoinFail : 'suppressJoinFailure=' Ponct ID Ponct;

xmlns : 'xmlns' Ponct ID Ponct ID Ponct ID Ponct;

importact : BEGIN_IMPO namesplocat importType (END_ACT | OPE END_IMPOR);

namesplocat : location namespace;

location : 'location=' Ponct ID Ponct;

namespace : 'namespace=' Ponct ID Ponct ID Ponct;

importType : 'importType='Ponct ID Ponct ID Ponct;

msgexchangs : BEGIN_MSGEXCHANGS (msgexchang)+ END_MSGEXCHANGS;

msgexchang : BEGIN_MSGEXCHANG name OPE END_MSGEXCHANG;

correlations : BEGIN_CORRLAS (correlation)+ END_CORRLAS;

correlation : BEGIN_CORRLA initiate OPE END_CORRLA;

initiate : 'initiate=' Ponct ID Ponct ;

correlationsset : BEGIN_CORRLASSET (correlationset)+ END_CORRLASSET;

correlationset : BEGIN_CORRLASET name OPE END_CORRLASET;

partnerlinks : BEGIN_PARTLINKS (partnerLink)+ END_PARTLINKS;

partnerLink : BEGIN_PARTLINK name partlinktyp (myrole)? (partnerRole)? (END_ACT | OPE END_PARTLINK);

partlinktyp : 'partnerLinkType=' Ponct ID Ponct ID Ponct;

myrole : 'myRole=' Ponct ID Ponct;

partnerRole : 'partnerRole=' Ponct ID Ponct;

vars : BEGIN_VARS (var)+ END_VARS;

var : BEGIN_VAR name messagetype (END_ACT | (initialization)? OPE END_VAR);

messagetype : ('messageType=' | 'type=') Ponct ID Ponct ID Ponct;

initialization : BEGIN_FROM partlink endpointref OPE END_FROM;

partlink : 'partnerLink=' Ponct ID Ponct ;

endpointref : 'endpointReference=' Ponct ID Ponct ;

actbpel : invoke | receive | reply | assign | whileact | repeatUntil | ifcond | flow | pick | sequence | scopeact | exitact | throwact | rethrow | compensate | foreach | waitact | emptyact | opacact | validateact;

# Basic activities

receive : BEGIN_REC name partlink porttypoperat variable (createinstance)? (messageexchange)? (END_ACT| OPE (correlations)? (documentation)? END_REC);

porttypoperat : ^(PORTOPDEF porttype operation ) | ^(PORTOPDEF operation porttype ) ;

createinstance : 'createInstance=' Ponct ID Ponct ;

messageexchange : 'messageExchange=' Ponct ID Ponct;

porttype : 'portType=' Ponct ID Ponct ID Ponct;

operation : 'operation=' Ponct ID Ponct ;

invoke : BEGIN_INVOK name partlink operation porttype inputvariable (outvariable)? (END_ACT | OPE (documentation)? END_INVOK);

inputvariable : 'inputVariable=' Ponct ID Ponct ;

outvariable : 'outputVariable=' Ponct ID Ponct ;

assign : BEGIN_ASSIGN validate name OPE (copy)+ END_ASSIGN;

copy : BEGIN_COPY from to END_COPY;

from : BEGIN_FROM (varpartfr)? (EXPRLANGUXPAT | EXPRLANGUXPATHINBPEL)?
(OPE (valcontentto)? (OPE)? END_FROM | END_ACT);

to : BEGIN_TO (varpartto)? (OPE (valcontentto)? (OPE)? END_TO | END_ACT);

varpartfr : variable part | part variable | expr | variable | partlink (endpointref)?;

varpartto : variable part | part variable | variable | partlink (endpointref)?;

valcontentto : expr | (idponct)+ | query ;

query : BEGIN_QUERY ID Ponct ID END_QUERY;

validate : 'validate=' Ponct ID Ponct;

variable : 'variable=' Ponct ID Ponct ;

part : 'part=' Ponct ID Ponct ;

expr : BEGIN_EXPR (idponct)+ END_EXPR;

reply : BEGIN_REP name partlink porttypoperat variable (faultname)? (messageexchange)?
(END_ACT | OPE (documentation)? (correlations)? END_REP);

faultname : 'faultName=' Ponct ID Ponct ID Ponct;

condition : BEGIN_CONDITION (expressionLanguage)? OPE '<![CDATA[' valconds
END_CONDITION;

valcond : ID Ponct ID (opeponct)+ ID Ponct ID | ID Ponct ID (opeponct)+ ID | ID (opeponct)+ ID
ID (opeponct)+ ID Ponct ID | ID Ponct Ponct ID Ponct | ID Ponct ID Ponct Ponct ID
Ponct;

idponct : ID (opeponct (ID)?)*;

opeponct : OPE | Ponct;

valconds: valcond (valcond)* ;

expressionLanguage : 'expressionLanguage=' Ponct ID Ponct ;

# Structured activities

repeatUntil : BEGIN_REPEATUNTIL name OPE (actbpel)* condition (documentation)?
END_REPEATUNTIL ;

whileact : BEGIN_WHILE name OPE condition (actbpel)* END_WHILE ;

ifcond : BEGIN_IF name OPE condition (actbpel)+ (elseifcond)* (elsecond)? (documentation)?
END_IF;

elsecond : BEGIN_ELSE (actbpel)+ END_ELSE;

elseifcond : BEGIN_ELSEIFCOND condition (actbpel)+ END_ELSEIFCOND;

flow : BEGIN_FLOW name OPE (actbpelfl)+ END_FLOW;

waitact: BEGIN_WAIT name OPE foralarm END_WAIT;

emptyact: BEGIN_EMPTY name OPE END_EMPTY;

opacact: BEGIN_OPAQACT name OPE END_OPAQACT;

validateact: BEGIN_VALIDACT name variablesact OPE END_VALIDACT;

variablesact : 'variables=' Ponct (ID)+ Ponct;

sequence : BEGIN_SEQ (name)? OPE (actbpel)* END_SEQ;

foreach : BEGIN_FOREACH parallel countername name (xmlns)? OPE startcountervalue
finalcountervalue (complcond)? scopeact (documentation)?
END_FOREACH;

complcond : BEGIN_COMPLETCOND branch END_COMPLETCOND;

branch : BEGIN_BRANCH succbranchonly OPE '<![CDATA[' (idponct)+ END_BRANCH ;

parallel : 'parallel=' Ponct ID Ponct;

countername : 'counterName=' Ponct ID Ponct;

expridcount : BEGIN_EXPR ID END_EXPR;

startcountervalue : BEGIN_STARTCOUNTVALUE (EXPRLANGUXPAT |
EXPRLANGUXPATHINBPEL)? OPE endstartcountvalue ;

endstartcountvalue : '<![CDATA[' ID END_STARTCOUNTVALUE | expridcount
END_STARTCOUNTVALUE2 ;

finalcountervalue : BEGIN_FINALCOUNTVALUE (EXPRLANGUXPAT |
EXPRLANGUXPATHINBPEL)? OPE endfinalcountvalue;

endfinalcountvalue : '<![CDATA[' ID END_FINALCOUNTVALUE | expridcount
END_FINALCOUNTVALUE2 ;

succbranchonly : 'successfulBranchesOnly=' Ponct ID Ponct;

pick : BEGIN_PICK name createinstance OPE (onmsg)+ (onalarm)? (documentation)?
END_PICK;

onmsg : BEGIN_MSG partlink operation porttype variable OPE (actbpel)+ END_MSG;

onalarm : BEGIN_ALARM scopeact foralarm END_ALARM;

foralarm : BEGIN_FOR (ID)+ END_FOR;

scopeact : BEGIN_SCOPE (name)? (isolated)? OPE (partnerlinks)? (vars)? (correlationsset)?
(msgexchangs)? (actbpel)+ (faulthandler)? (compensatehandl)? (terminhandl)?
(eventhandler)? (documentation)? END_SCOPE ;

isolated : 'isolated=' Ponct ID Ponct ;

# Faults activities

faulthandler : BEGIN_FAULTHANDLER (catchact)+ (catchactall)? (documentation)?
END_FAULTHANDLER;

catchact : BEGIN_CATCH (faultname)? (faultvariable)? OPE (actbpel)* (documentation)?
END_CATCH;

catchactall : BEGIN_CATCHALL (actbpel)* (documentation)? END_CATCHALL;

faultvariable : 'faultVariable=' Ponct ID Ponct ;

compensatehandl : BEGIN_COMPENSATHANDL (actbpel)* (documentation)?
END_COMPENSATHANDL;

terminhandl : BEGIN_TERMINHANDL (actbpel)* (documentation)? END_TERMINHANDL;

eventhandler : BEGIN_EVENTHANDL (onevent)+ (onalarm)? (documentation)?
END_EVENTHANDL;

onevent : BEGIN_ONEVENT partlink operation (messageexchange)? OPE (correlationsset)?
scopeact (documentation)? END_ONEVENT;

exitact : BEGIN_EXIT name OPE (documentation)? END_EXIT;

throwact: BEGIN_THROW name faultname faultvariable OPE (documentation)? END_THROW;

rethrow : BEGIN_RETHROW name OPE (documentation)? END_RETHROW;

compensate : BEGIN_COMPENSATE name OPE (documentation)? END_COMPENSATE;

compensatescope : BEGIN_COMPENSATESCOP name OPE (documentation)?
END_COMPENSATESCOP;

documentation : BEGIN_DOC (source)? (language)? OPE (expr)? END_DOC;

source : 'source='Ponct ID Ponct ;

language : 'language='Ponct ID Ponct ;

name : 'name=' Ponct ID Ponct;