

AO4AADL grammar

Sihem Loukil, Slim Kallel, Bechir Zalila, and Mohamed Jmaiel

July 2011

We present here all the grammar rules of our proposed language AO4AADL.

Aspect_Annex ::= { Aspect_Expression }+

Aspect_Expression ::= aspect { *Aspect_Identifier* } {
 [Aspect_Precedence ;]
 [Components_appliesd_to ;]
 { Pointcut_Specification ; }+
 { Advice_Specification }+
}

Aspect_Precedence ::= **precedence** *Aspect_Identifier* { , *Aspect_Identifier* }*

Components_appliesd_to ::= **applies to** ListComponents

ListComponents ::= Component { ,Component }*

Component ::= Component_Category *Component_Identifier*

Component_Category ::= **thread** | **process** | **subprogram**

Pointcut_Specification ::= **pointcut** *Pointcut_Identifier* ([ParamList]) :
 Pointcut_Expression

ParamList ::= Parameter_Specification { ,Parameter_Specification }*

Parameter_Specification ::= *Parameter_Identifier* : *Type_Identifier*

Identifier ::= Character { CharacterOrNumeral }* ;

Character ::= **a .. z** | **A .. Z**

CharacterOrNumeral ::= Numeric_Literal | Character | _

Numeric_Literal ::= **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9**

Pointcut_Expression ::= Pointcut_Primitive | (Pointcut_Expression)
 | Pointcut_Expression && Pointcut_Expression
 | Pointcut_Expression || Pointcut_Expression

Pointcut_Primitive ::= Call | Execution | Args

Call ::= **call** Callee

Execution ::= **execution** Callee

Callee ::= **subprogram** (*Subprogram_Identifier*
 ([*Subprogram_Parameter_Types*]))
 | **import** (*Input_Port_Identifier* ([*Data_Type*]))
 | **output** (*Output_Port_Identifier* ([*Data_Type*]))
 | **inoutport** (*InOutput_Port_Identifier* ([*Data_Type*]))

Subprogram_Parameter_Types ::= Type { ,Type }*

Data_Type ::= Type

Type ::= .. | *Type_Identifier*

Subprogram_Identifier ::= *Identifier* | *

Input_Port_Identifier ::= *Identifier* | *

Output_Port_Identifier ::= *Identifier* | *

InOutput_Port_Identifier ::= *Identifier* | *

Args ::= **args** (Arguments)

Arguments ::= Argument { ,Argument }*

Argument ::= .. | *Argument_Identifier* { ,*Argument_Identifier* }*

Advice_Specification ::= **advice** *Advice_Declaration* : *Pointcut_Reference*
 Advice_Action

Advice_Declaration ::= *Before_Advice* | *After_Advice* | *Around_Advice*

Before_Advice ::= **before** ([*Paramlist*])

After_Advice ::= **after** ([*Paramlist*])

Around_Advice ::= **around** ([*Paramlist*])

Pointcut_Reference ::= *Pointcut_Identifier* ([*Parameters*])

Parameters ::= *Parameter_Identifier* { ,*Parameter_Identifier* }*

Advice_Action = {
 [*Variables_Declaration*]
 [*Initialisation*]
 Action +

}

Variables_Declaration ::= **variables** { { Variable }+ }

Variable ::= *Variable_Identifier* { , *Variable_Identifier* }* : *Type_Identifier* ;

Initialisation ::= **initially** { Assignment + }

Action ::= Basic_Action
 | If_Statement
 | For_Statement
 | While_Statement

If_Statement ::= **if** (Behavior_Expression) {
 { Action }+
 }
 { **else if** (Behavior_Expression) {
 { Action }+
 }
 }
 }*
 [else {
 { Action }+
 }
]]

For_Statement ::= **for** (*Loop_Variable_Identifier* **in** Integer_Range) {
 { Action }+ }

Integer_Range ::= Arith_Expression .. Arith_Expression

While_Statement ::= **while** (Behavior_Expression) { Action + }

Basic_Action ::= Assignment
 | Communication
 | Timed_Actions
 | Proceed_Action

Assignment ::= Reference_Expression := Behavior_Expression

Reference_Expression ::= *Loop_Variable_Identifier*
 | *Variable_Identifier*
 | *Parameter_Identifier*
 | *Argument_Identifier*

Behavior_Expression ::= Disjunction { **or** Disjunction } *
 | 'Character'

| ” String_Literal ”

Disjunction ::= Not_Conjunction { **and** Not_Conjunction } *

Not_Conjunction ::= [**not**] Conjunction

Conjunction ::= Arith_Expression [(<|≤|=|>|≥|!=) Arith_Expression]
| Boolean_Literal

Arith_Expression ::= Add_Expression { (+ | -) Add_Expression } *

Add_Expression ::= Basic_Expression { (* | /) Basic_Expression } *

Basic_Expression ::= Constant_Expression | Reference_Expression

Constant_Expression ::= Numeric_Literal
| *Port_Identifier* ' **count**

Communication ::= *Required_Subprogram_Identifier* ! [(Parameter_Profile)]
| *Output_Port_Identifier* ! (*Data_Identifier*)
| *Input_Port_Identifier* ? (*Data_Identifier*)

Boolean_Literal ::= **true** | **false**

Timed_Actions ::= **computation** (Behavior_Time [, Behavior_Time]);
| **delay** (Behavior_Time [, Behavior_Time]);

Behavior_Time ::= Arith_Expression *Unit_Identifier*

Proceed_Action ::= **proceed** ([Parameter_Profile]);

Parameter_Profile ::= Parameter { ,Parameter }*

Parameter ::= *Parameter_Identifier* | Behavior_Expression

Data_Identifier ::= *Parameter_Identifier* | Behavior_Expression