# Event Definition Language

The basic elements of the event definition language include:

* Event definition
* Subscribe statement

For each of the event definition, it can include:

* Attribute list
* Event filter (specified by using a where clause)
* Event composition (specified by using an on clause)

## BNF

The simplified BNF is as follows (start from non-terminal ‘goal’). Please note that the identifiers written in all capital letters are terminals while others are non-terminals

goal ::= type\_declarations\_opt;

type\_declarations\_opt ::= type\_declarations|;

type\_declarations ::= type\_declaration | type\_declarations type\_declaration;

type\_declaration ::= event\_declaration |subscribe\_statement;

event\_declaration ::= EVENT IDENTIFIER event\_body composite\_body\_opt filter\_body\_opt;

event\_body ::= LBRACE field\_declarations\_opt RBRACE

filter\_body ::=WHERE LBRACE conditional\_expression RBRACE

composite\_body ::= LBRACE field\_declarations\_opt RBRACE

## Built-in Functions

The event definition language supports the following built-in functions:

* Average(): calculate the average
* Sum(): calculate the sum
* Count(): count the total number of events

## Examples

A simple event which detects the light change:

Event SimpleLight {

Int light=System.light;

Int time=System.time;

} where {

Light >30

}

A composite event made from two simple events described above:

Event CompositeLight {

Int light=light1.light;

Int time=light2.light;

} on {

SimpleLight light1;

SimpleLight light2;

} where {

Light2.time-light1.time>3

}

An event which makes use of the built-in function to read the average light reading:

Event AverageLight {

Int light=average(System.light);

} where {

Light >30

}

An event which counts the car number in an area:

Event CarCount {

Int count=count(car)

} on {

CarEvent car;

} where {

Car.locationX==1 && car.locationY==3

}