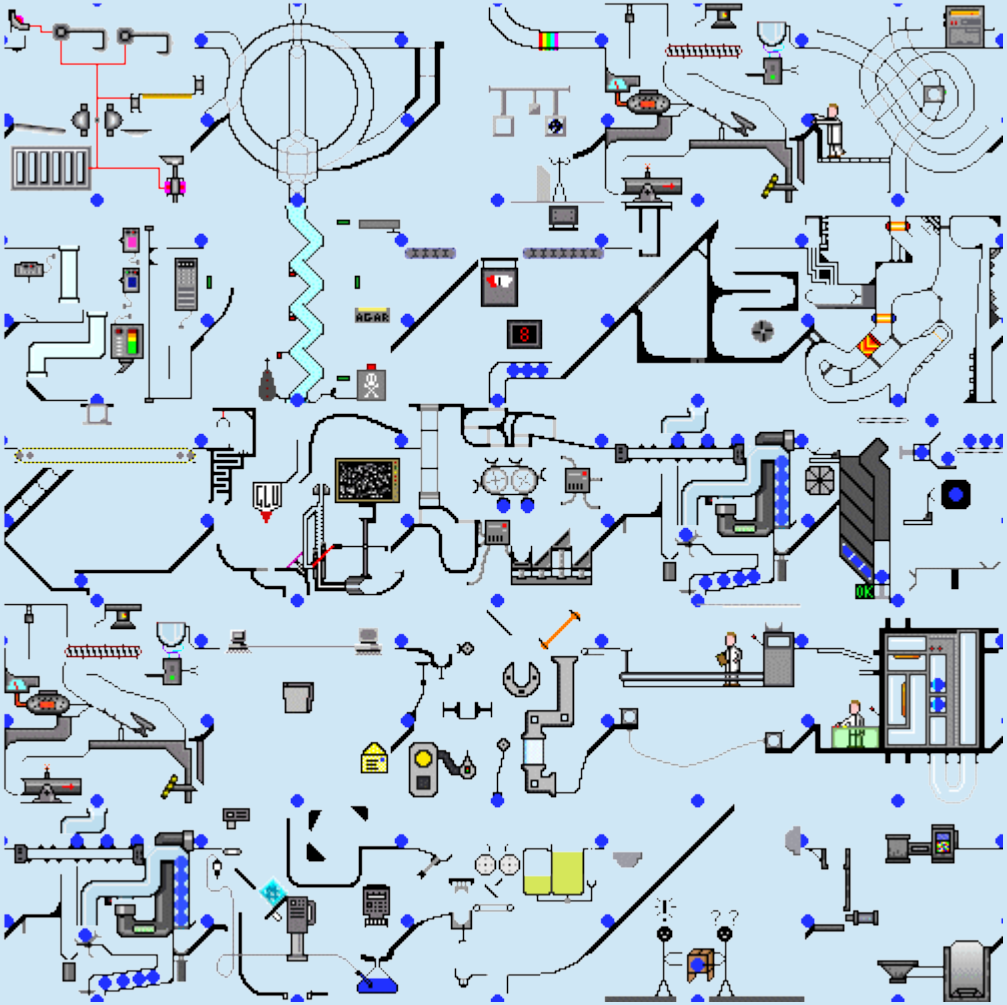
Dclare for MPS

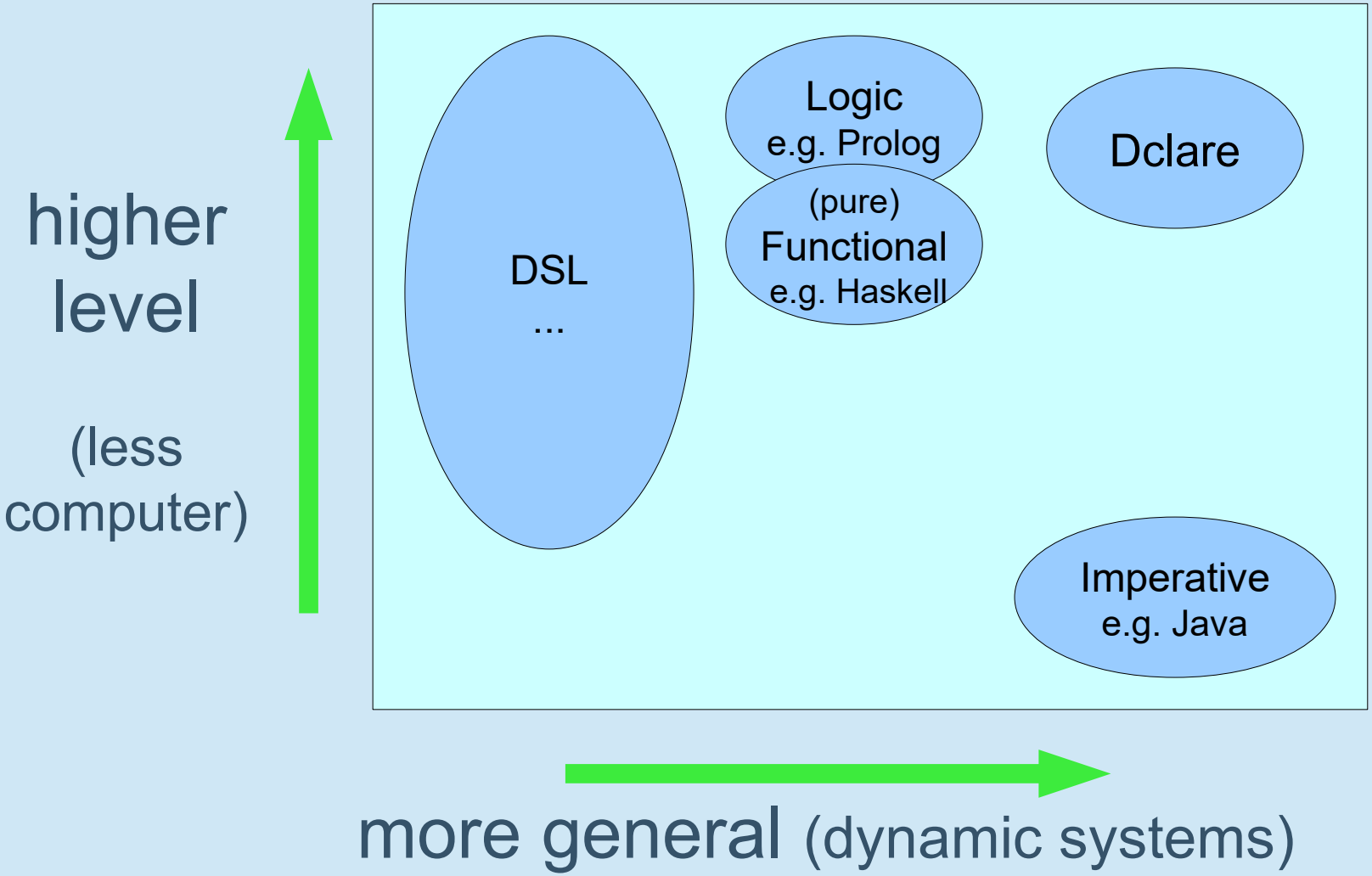
A declarative base language for MPS

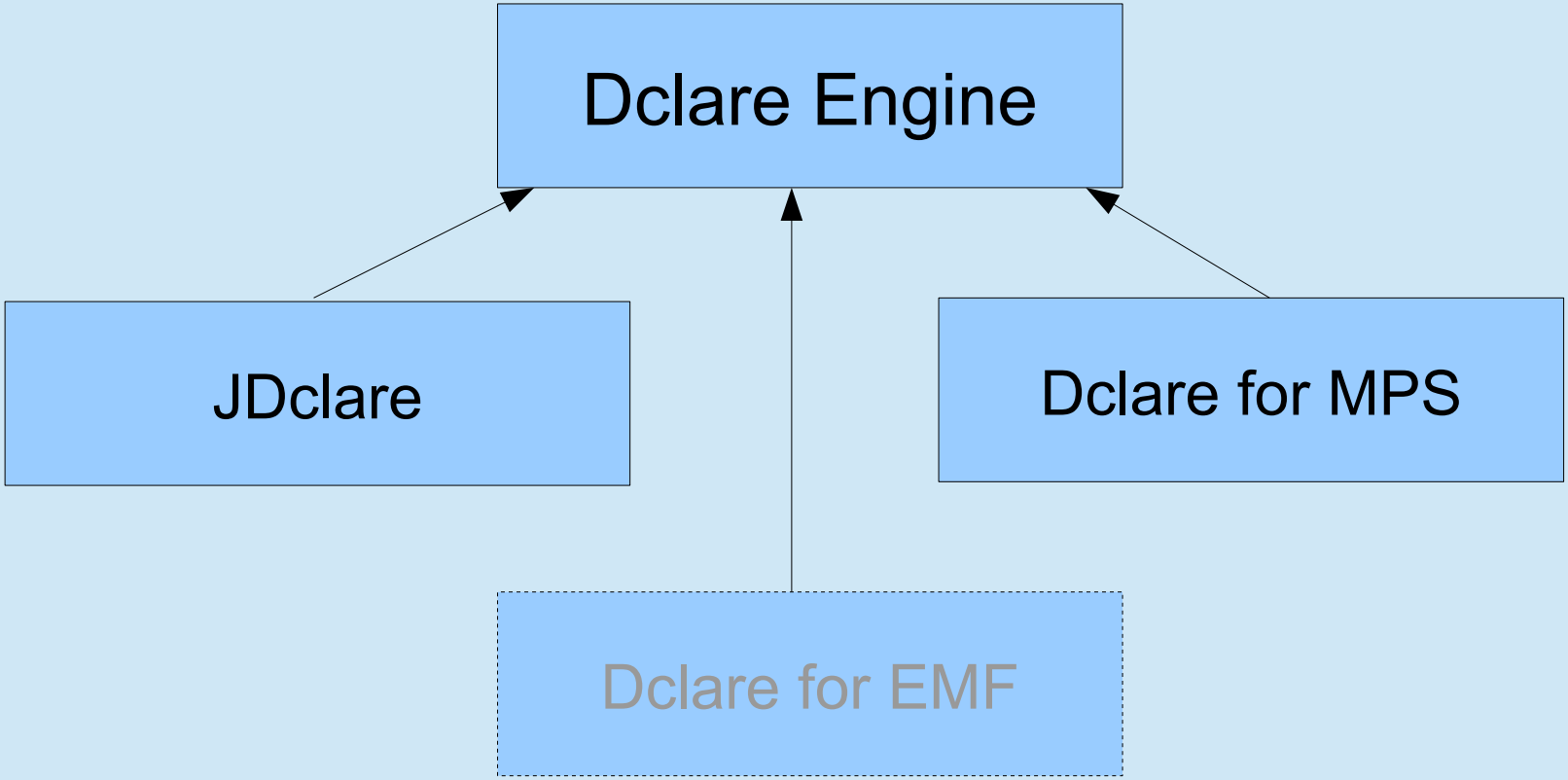
https://github.com/ModelingValueGroup/DclareForMPS

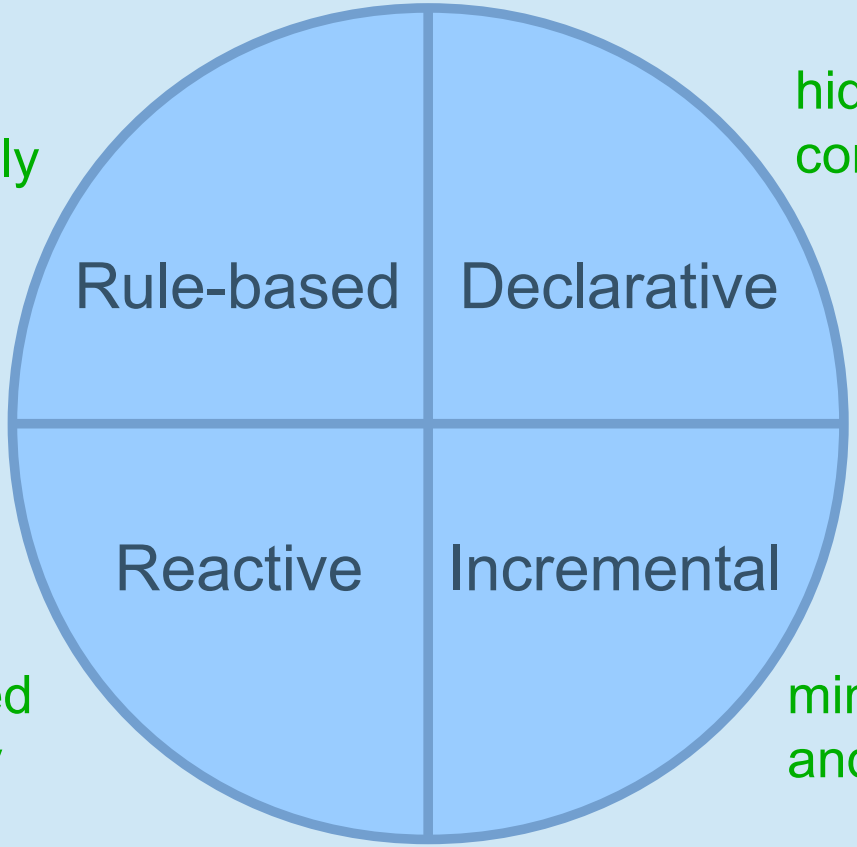
Declare

MODELING VALUE GROUP
SOFTWARE FACTORIES







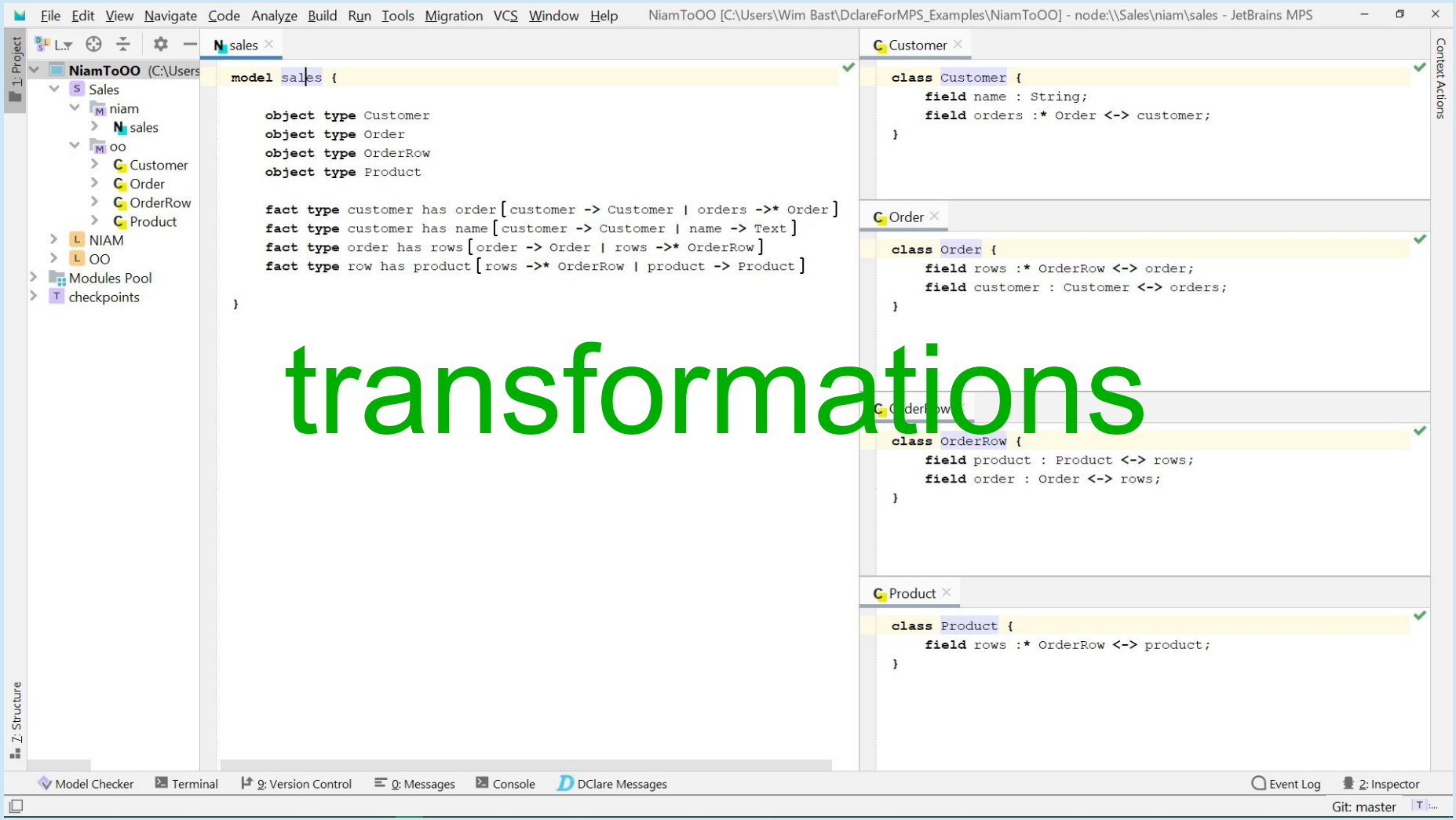


all invariants hold eventually

hides threading and control completely

rules triggered automatically

minimal triggering and changes



transformations

Declare

MODELING VALUE GROUP
SOFTWARE FACTORIES

CellLogic

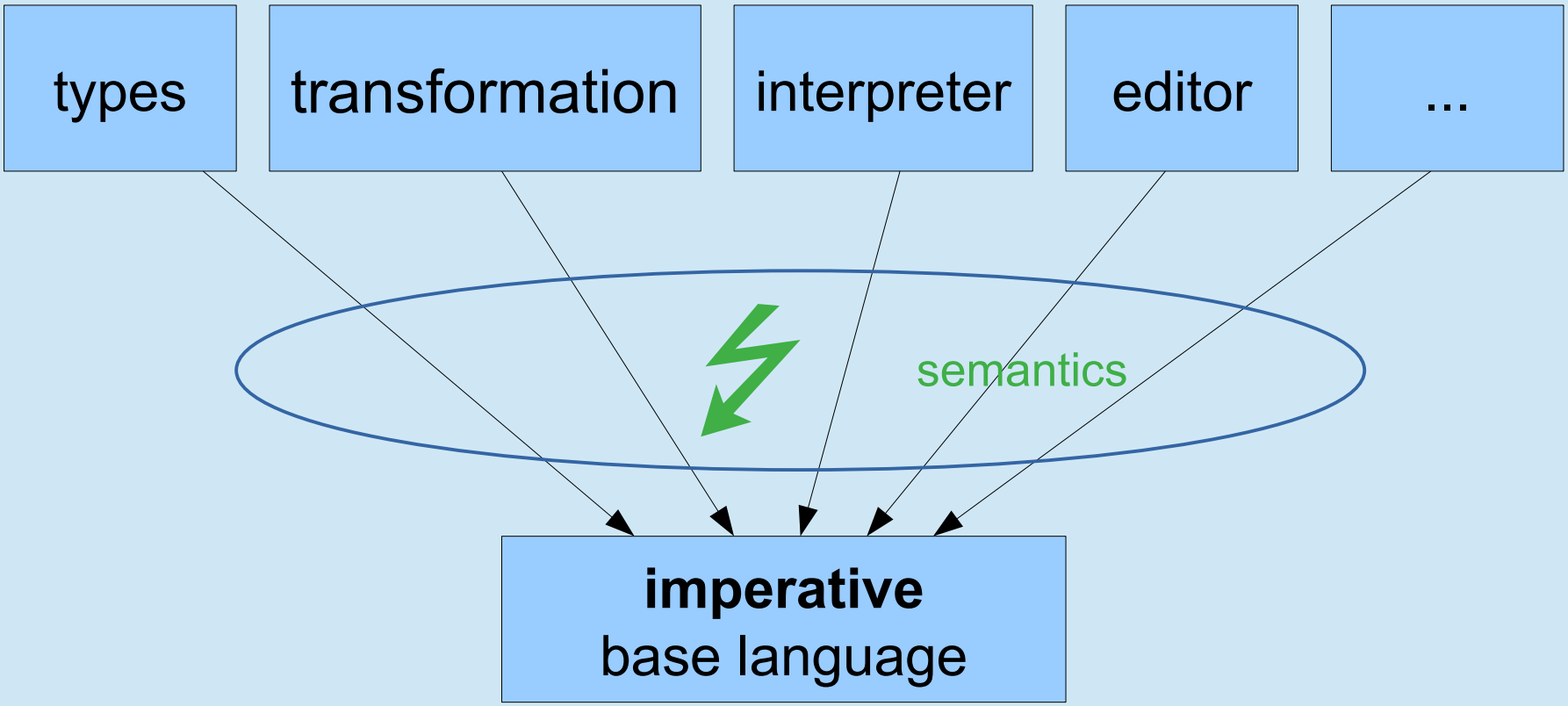
```
struct ruleset CellLogic {

  // attributes

  identifying node<Cell> cell;

  identifying list<int> filling;

  node<Sudoku> sudoku := cell.sudoku;

  list<struct<Context>> contexts := [
    sudoku.rows[cell.row],
    sudoku.columns[cell.column],
    sudoku.blocks[cell.block]];

  list<set<struct<CellLogic>>> others := contexts.select({~c =>
      c.cells.where({~c => c.cell :ne: cell && c.filling :eq: filling }).toSet });

  set<int> takenByOthers := others.selectMany({~o => o; }).where({~c => c.possible.size = 1; })
      select({~c => c.possible.first; });

  set<int> possible := cell.isSet ? new singleton<int>(cell.value) :
      possible.size != 1 ? sudoku.values.where({~v => !takenByOthers.contains(v); }) : possible;

  list<set<int>> possibleByOthers := others.select({~o => o.possible.toSet; });

  // rules

  if (possible.size > 1) {
    foreach po in possibleByOthers {
      set<int> must = possible.where({~v => !po.contains(v); }).toSet;
      if (must.isNotEmpty) {
        possible := must;
      }
    }
  }
```

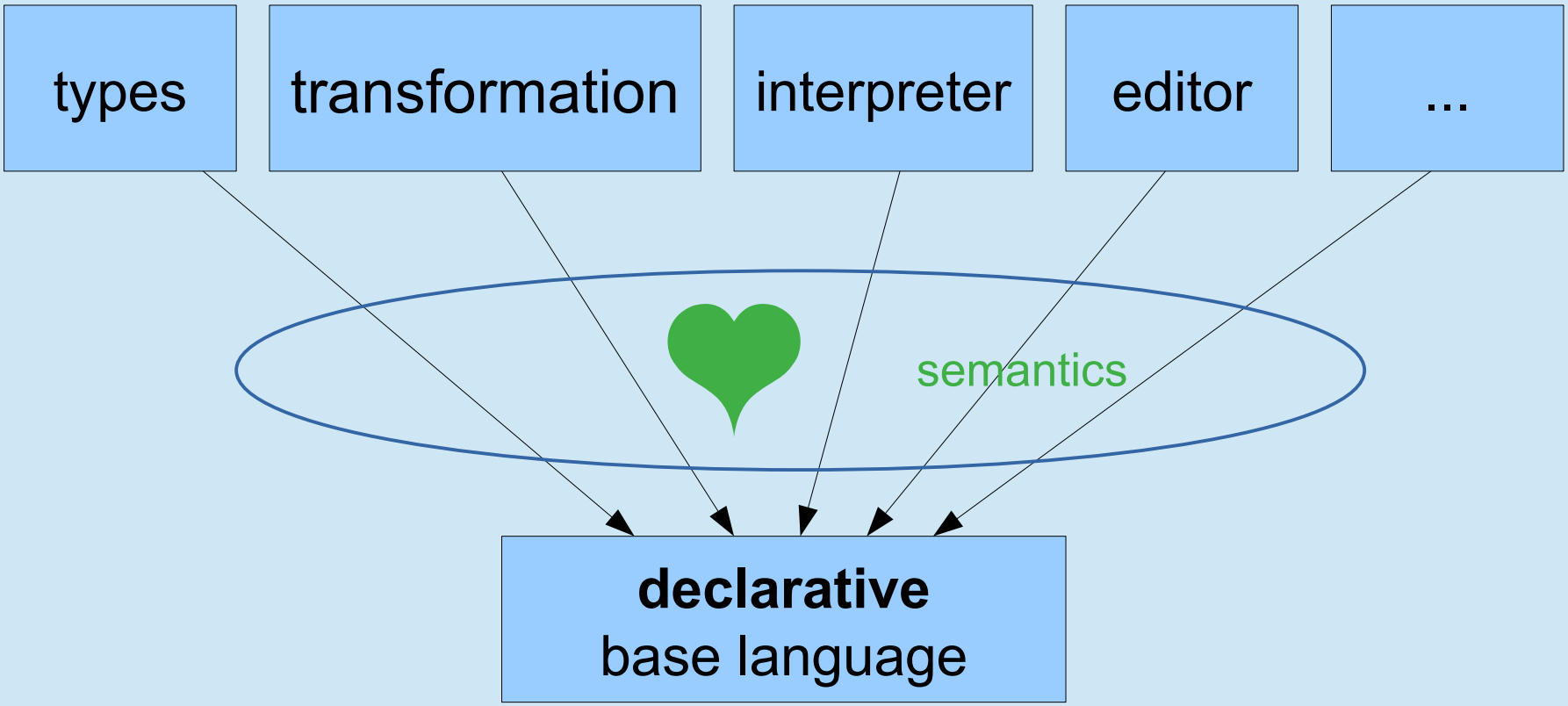
sudoku31

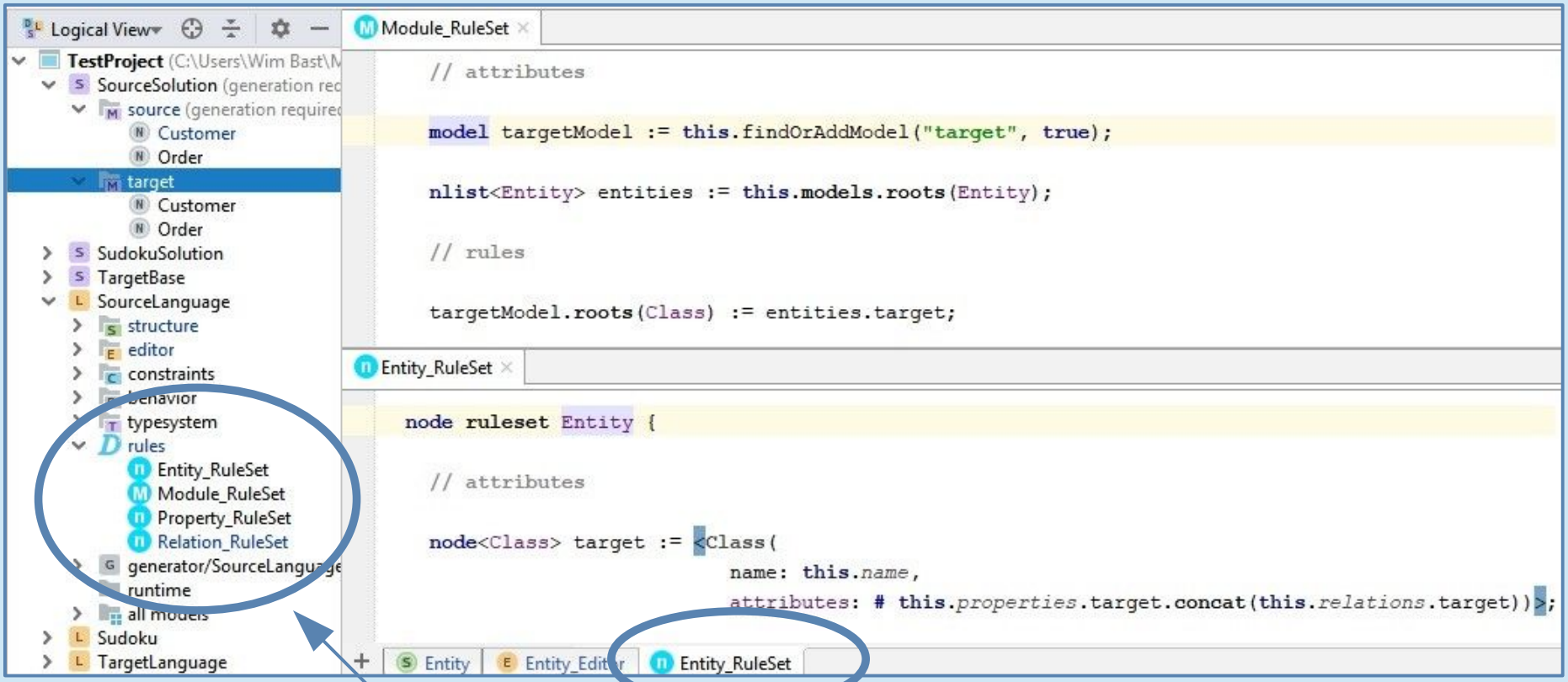
sudoku31 base size 3

5						8		
4			2		8		9	
				7		6		
		2						
	3	5			9			
6		7	5	3				
2	5	9	1	6	3	4	7	8
		4	8	9				2
		8	4			1		9

interpreters

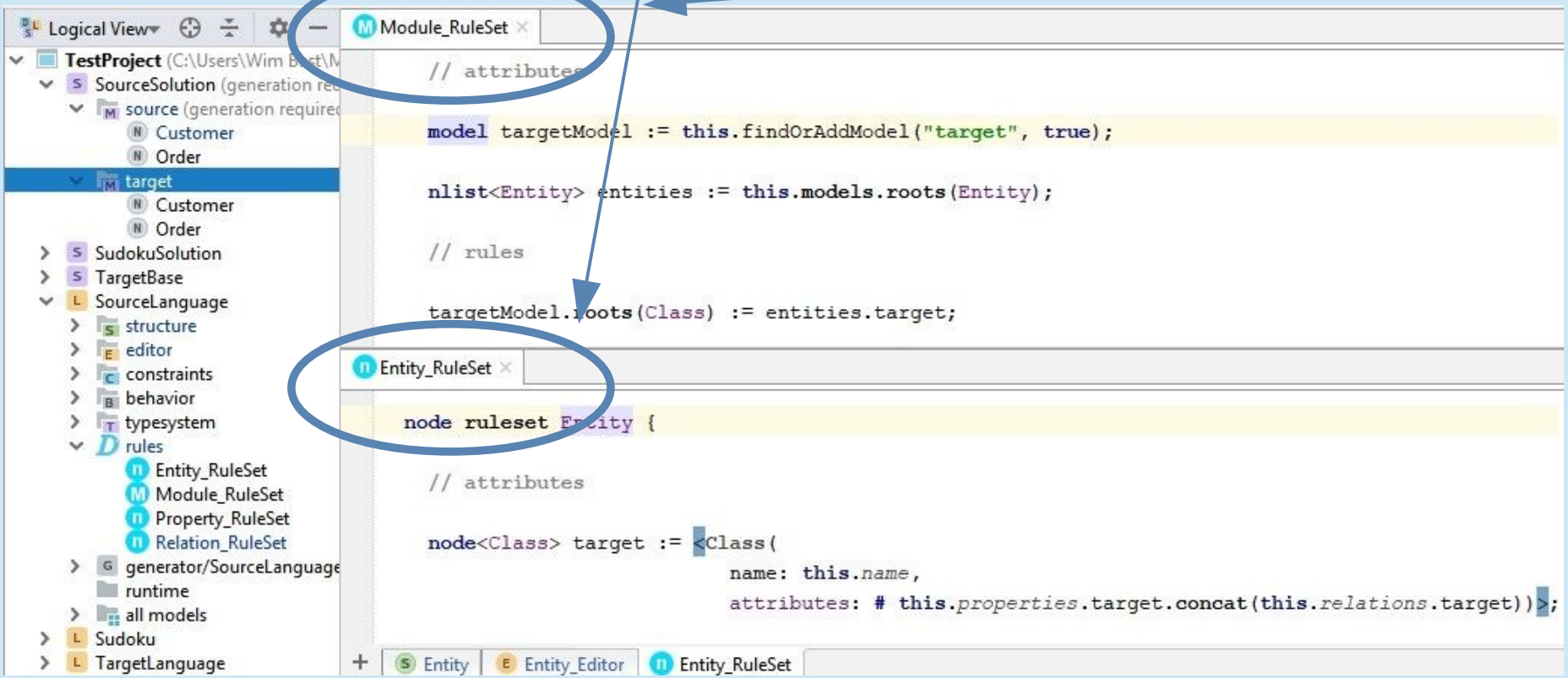




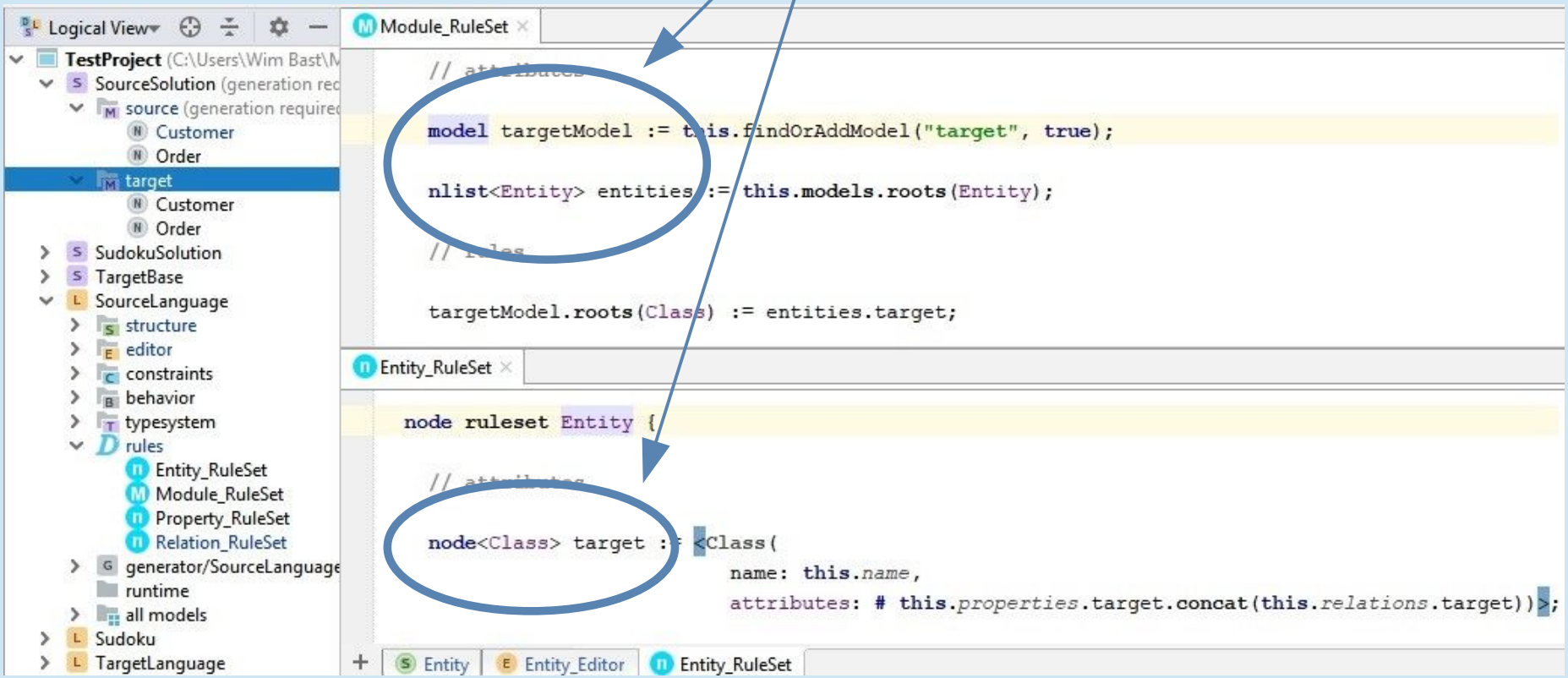


complete language aspect

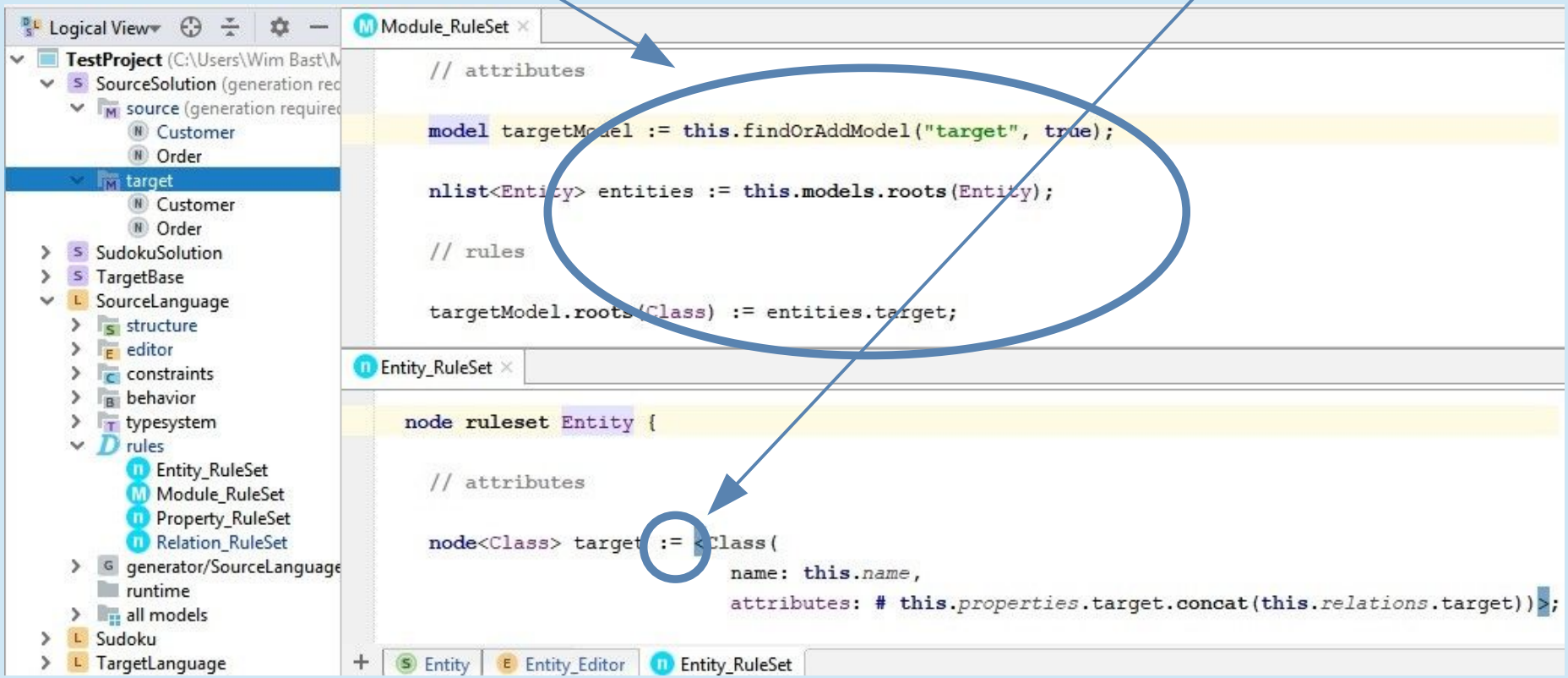
rule-sets (on nodes, models, modules, structs)

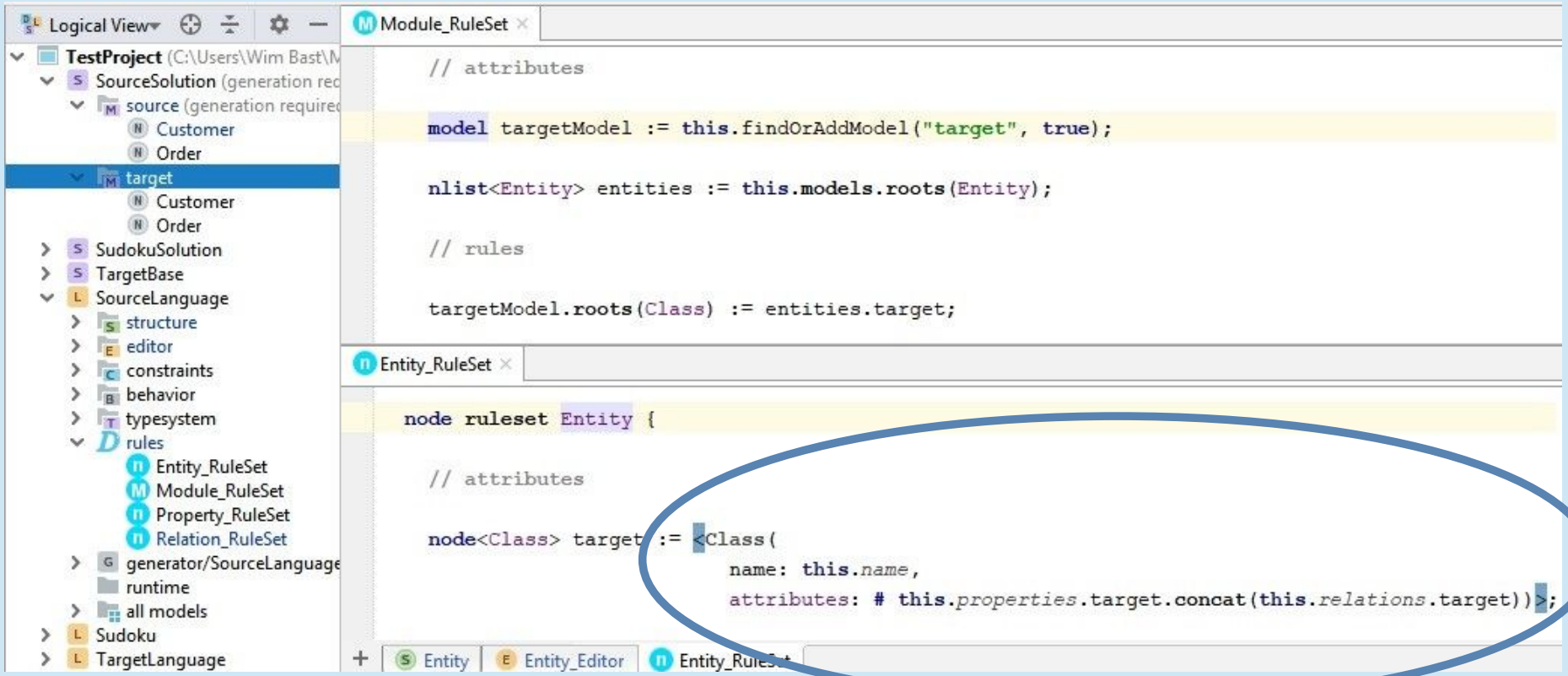


attributes of any type



Rules: base-language & equations





incremental quotations (light or not) and copy

DEMO

Dclare for MPS

Q & A

https://github.com/ModelingValueGroup/DclareForMPS