SysML MODELING RULES FOR SysML TO B

File name: 2014_12_04_SYSMLMODELINGRULES-FOR-SYSMLTOB_MFR14-ARC-839--MFR14-ECD-530.DOCX
Version: 0.2
Date: 2014-12-04
Author(s): Thomas Bardot

Classification Status: Non confidential
Export Control Status: NLR
Edition Status: Draft

Approved by:	SPT	Date:	2014-11-18
Diffusion list:	MELCO, openETCS, Public		

Revision history

Version	Date (DD/MM/YYYY)	Description	Editor(s)
0.1	2014-11-04	Draft	TBD
0.2	2014-11-18	minors corrections	TBD – DME
0.2	2014-12-04	openETCS release	DME

Table of Content

1 PURPOSE

This document is under openETCS license.

This document takes place in the context of the automatic translation of a SysML model in a B model architecture.

Because the SysML language offers many diagrams and elements, a SysML subset has been defined in order to help the modeling task (see [1]). It is also necessary to define modeling rules with the objective that the model can be interpreted by the SysML to B translation tool.

Therefore, the purpose of this document is to define the modeling rules applicable on a SysML model in order to translate it into B code.

2 REFERENCE DOCUMENTS

Tag	Document
[1]	D2_4 *Definition of the methods used to perform the formal description* https://github.com/openETCS/requirements/blob/master/Reference/D2_4.pdf?raw=true
[2]	"A Practical Guide to SysML", S. Friedenthal, A. Moore, R. Steiner
[3]	SysML standard 1.3 http://www.omg.org/spec/SysML/1.3/. Note that we use the deprecated features flow port and flow specification.

3 INTRODUCTION

3.1 CONTEXT

The modeling rules defined in this document permit to make a SysML model understandable for the SysML to B translator tool.

Therefore, these modeling rules are made in accordance with the exported constraints of the model translator and with the choice of the SysML subset that can be used (see [1]).

Moreover, because the SysML language relies upon the SysML modeler used, this document is made on the hypothesis that the SysML modeler is Papyrus for Eclipse.

The SysML language is defined by the OMG group (see [3]) but it is recommended to also refer to the reference book "A Practical Guide to SysML" by Friedenthal *et al*. (see [2]).

3.2 METHODOLOGY

Because we want that the modeling rules cover all the SysML subset, we analyze separately each of the selected diagrams and each of their owned selected elements by the document which defined a SysML subset for the modeling activity of the OpenETCS project (see [1]). Additional SysML elements that are used by the SysML to B translator tool are analyzed in this document.

For each SysML element, we describe:

- How the element is interpreted by the SysML to B translator tool;
- The modeling rules associated to this element.

4 MODEL

4.1 MODEL STRUCTURE

RULE_1 The SysML model shall have a "Model" UML-type element at the top of it hierarchical tree.

4.2 GLOBAL NAMING CONVENTION

It is recommended to apply a naming convention in order to increase the readability of a model. The document [1] defined a naming convention which is applicable to the OpenETCS project.

RULE_2 None of the SysML element can have the same name than a B statement (OPERATIONS, VAR, ANY, IF, BEGIN, etc.)

5 PACKAGE DIAGRAM

5.1 COMMENT NOTE

This element is not used by the SysML to B translator tool.

5.2 PACKAGE NODE

Packages are seen as containers of other SysML elements, they contain the model (see Figure 1). It is recommended to use them in order to improve the model readability but the SysML to B translator does not translate them.

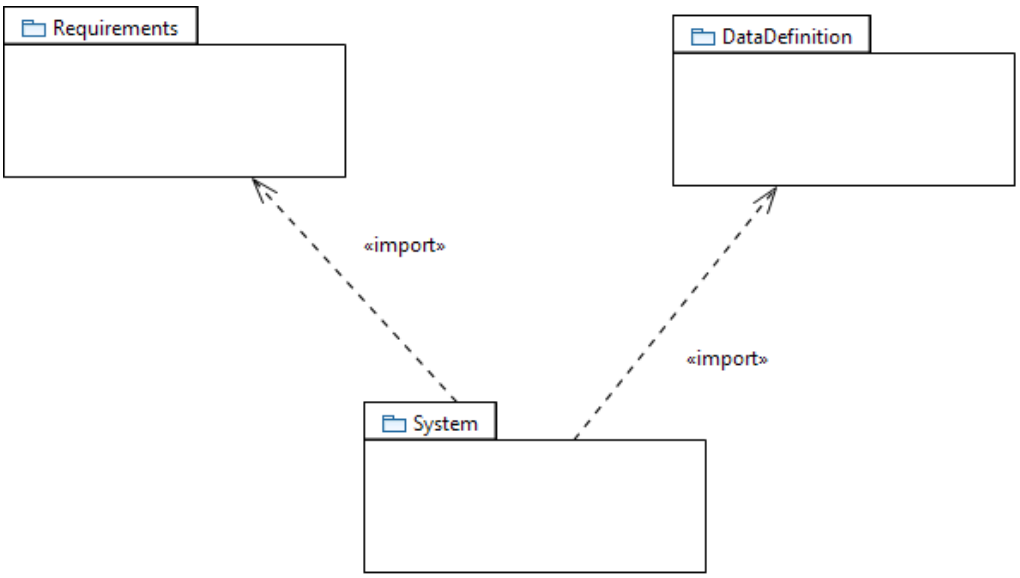

Figure 1: typical model organization, all the model elements are contained by these three packages

5.3 PACKAGEABLE ELEMENT NODE

Packageable Element node defines the model elements that can be contained in packages. The model elements contained in a package are those that are included in this package in the model's hierarchy (model explorer view in Papyrus).

5.4 IMPORT PATH

This element is not used by the SysML to C translator tool.

6 BLOCK DEFINITION DIAGRAM

6.1 BLOCK NODE

Blocks are interpreted as generic definition of functions which can have local variables, inputs and outputs and constraints. An instance of a block is translated into a B module by the SysML to B translator tool.

A block instance is defined with Instance Specifications element (see §6.12).

RULE_3 All the elements owned by a block (flow ports and properties) shall have a different name.

6.2 FLOW SPECIFICATION NODE

A flow specification node can be used to type a flow port. It can be seen as a data structure but it is translated by the SysML to B translator tool into several variables. The SysML to B translator tool will create one variable per Flow Specification field.

An example of flow specification nodes is shown by the Figure 2.

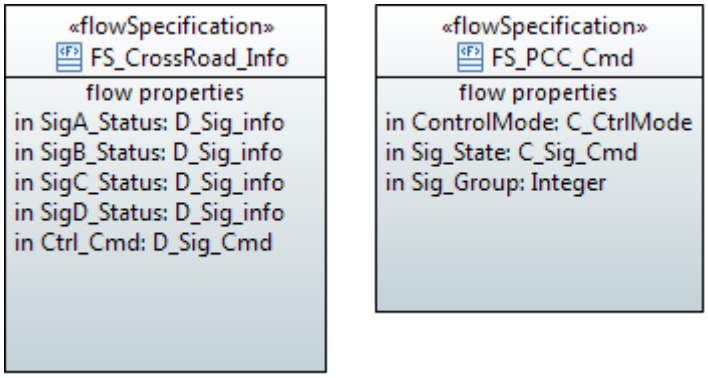

Figure 2: example of Flow Specification nodes that define *FS_CrossRoad_Info* and *FS_PCC_Cmd*. *FS_CrossRoad_Info* is composed of five flow properties which are typed with Data Type nodes (*D_Sig_Info* and *D_SigCmd* are Data Type nodes). *FS_PCC_Cmd* is composed of two flow properties typed with enumerated type (*ControlMode* and *Sig_State*) and one flow property typed with primitive type (*Sig_Group*).

RULE_4 A flow specification attribute can only be defined with a flow property node.
RULE_5 All the flow properties of a flow specification shall have a different name.

6.3 FLOW PROPERTY NODE

A flow property node is interpreted by the SysML to B translator tool as a member of a flow specification structure.

The Figure 2 shows an example of flow specification nodes which contain flow properties members.

RULE_6 A flow property node shall be associated to one flow specification node.
RULE_7 All the flow property nodes that belong to the same flow specification node shall have a different name.
RULE_8 All the flow property node shall be typed with a primitive type, a DataType or an enumeration node.

RULE_9 A flow property typed with a primitive type shall be initialized with a default value specified with an "Opaque Expression" (see §6.11).

RULE_10 A flow property typed with a DataType can be initialized with a default value specified with an "Opaque Expression"(see §6.11).

6.4 ENUMERATION NODE

A enumeration node is translated by the SysML to B translator tool into an enumerated set.

An example of enumeration nodes is shown by the Figure 3.

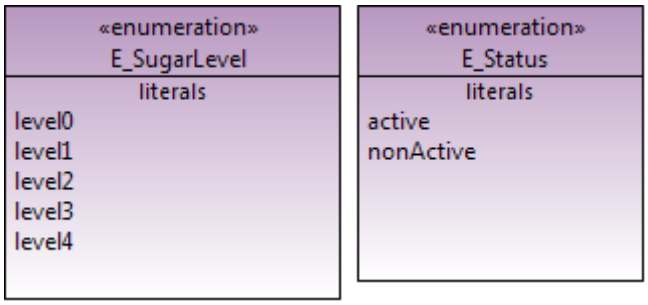

Figure 3: example of enumeration nodes that define *E_SugarLevel* and *E_Status*. *E_SugarLevel* is composed of five enumeration literals: *level0 to level4*. *E_Status* is composed of two enumeration literals: *active and nonActive*.

RULE_11 All the enumeration nodes of the model shall have a different name.

6.5 ENUMERATION LITERAL NODE

An enumeration literal node is translated by the SysML to B translator tool to an element of a set.

The Figure 3 shows an example of enumeration nodes which contain enumeration literal members.

RULE_12 An enumeration literal node shall be associated to an enumeration node.

RULE_13 The enumeration literals from any enumeration shall have different names, even if they belong to different enumerations.

RULE_14 An enumeration literal shall have a different name than all the flow ports and block properties

6.6 PRIMITIVE TYPE NODE

RULE_15 A SysML element which is typed with the UML Primitive Type "Boolean" is translated to a B variable with the "BOOL" type.

RULE_16 A SysML element which is typed with the UML Primitive Type "Integer" is translated to a B variable with the "INT" type.

RULE_17 A SysML element which is typed with the UML Primitive Type "String" is translated to a B variable with the "STRING" type.

RULE_18 A SysML element which is typed with a UML Primitive Type that is different from "Boolean" or "Integer" or "String" is translated to a B variable typed with the same name than the UML Primitive Type.

6.7 DATATYPE

A DataType can be seen as a data structure which types a SysML element but it is translated by the SysML to B translator tool into several variables. The SysML to B translator tool will create one variable per DataType attribute.

An example of DataType nodes is shown in the Figure 4.

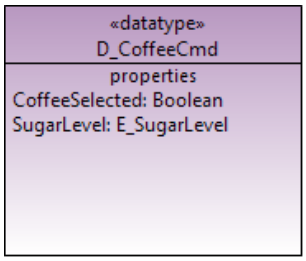

Figure 4: example of a dataType node that defines *D_CoffeeCmd*. *D_CoffeeCmd* is composed of two property nodes: *CoffeeSelected* and *SugarLevel*.

RULE_19 Every DataType element shall have a different name than the other "DataType" elements.
RULE_20 All the property nodes of a DataType node shall have a different name.

6.8 PROPERTY NODE

When it is associated to a block, a property node is interpreted by the SysML to B translator tool as a local variable of the function represented by the block.

When it is associated to a DataType, a property node is interpreted by the SysML to B translator tool as a member of the DataType structure.

The Figure 4 shows an example of a data type node which is composed of two property nodes.

RULE_21 A property node shall be typed with a Primitive Type node, a DataType node or a Enumeration node.
RULE_22 A property node typed with a Primitive Type node or a Enumeration node shall be initialized with a default value specified with an "Opaque Expression" (see §6.11).
RULE_23 A property node shall not be static ("Is static" property shall be set to False, which is the default value).

6.9 COMPOSITE ASSOCIATION PATH

A composite association path indicates that a block is composed of part properties typed with other blocks (see §7.1). It is not mandatory to create composite association path in order to create a part property: part properties can be created without using these paths, it can sometimes improve the understanding of a complex model.

In the example depicted in the Figure 5, the block *Controller* is composed of four parts, respectively typed with *SelectorController, TemperatureController, CoffeeMaker* and *Timer*. One composite association path has to be created for each part, even when the parts are typed with the same block (which it is not the case in the example).

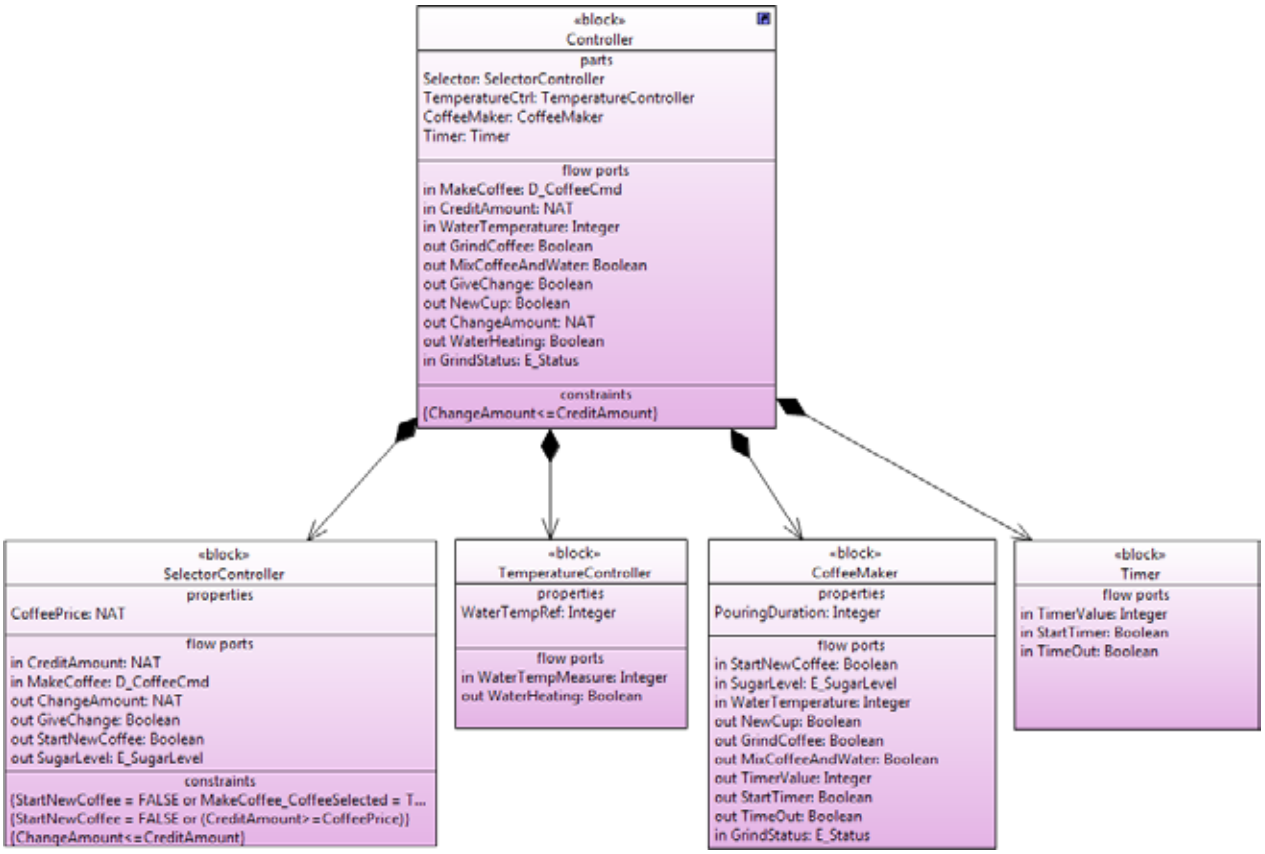

Figure 5: example of a composite association path

RULE_24 The directed composition association shall be used in order to represent a composite association path. Thus a part property will be created only in the owning block and no reference to the owning block will be create in the inner block.

RULE_25 The model shall be composed as a tree, then two blocks cannot be part of each other (see Figure 6).

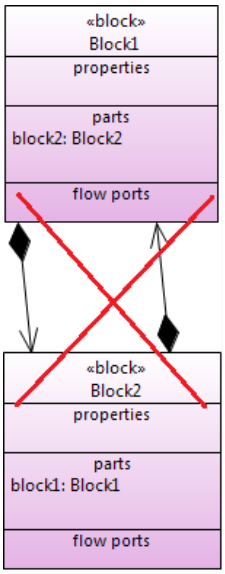

Figure 6: this example show a hierarchy that is not a tree, which is forbidden

6.10 CONSTRAINT NODE (FOR BLOCK OUTPUT INVARIANT)

When it belongs to a block, a constraint node is interpreted by the SysML to B translator tool as a logical expression which define an invariant on the inputs, outputs (see §7.3) and local variables (see §6.8) of a block.

The SysML to B generator will generate the invariant for each instance of the block.

An example of a constraint is shown on the Figure 7.

RULE_26 A constraint node shall be specified with an "Opaque Expression";

RULE_27 The opaque expression of a constraint node shall be a logical expression written in B language.

RULE_28 The variables used by the logical expression can only be the flow ports or property nodes of the owner block.

RULE_29 The name of the variable used by the logical expression shall be identical to the name of the corresponding flow port or property node.

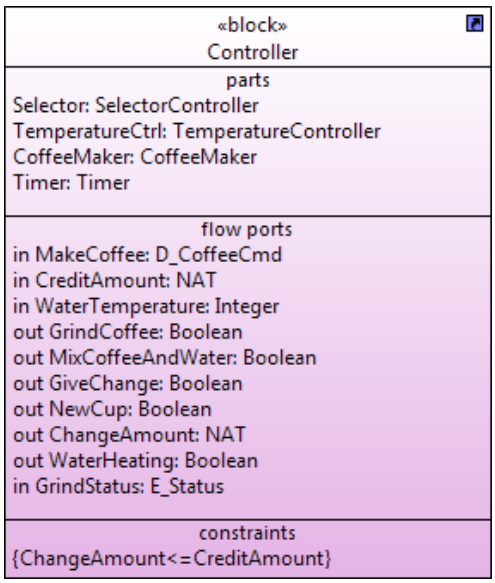

Figure 7: a constraint is added to a block. The constraint defines an invariant on an input (*CreditAmount*) and an output (*ChangeAmount*) of the block.

6.11 OPAQUE EXPRESSION (FOR DEFAULT VALUE)

When used to specify a default value, an opaque expression is translated by the SysML to B translator tool to one or more statements of "INITIALISATION" clause of B implementation.

An example of such opaque expression is shown on the Figure 8. The language parameter of the Opaque Expression is not verified by the SysML to B translator tool. Thus, in this example, the language is set to "C" even if we use the B language, it won't change anything on the translation.

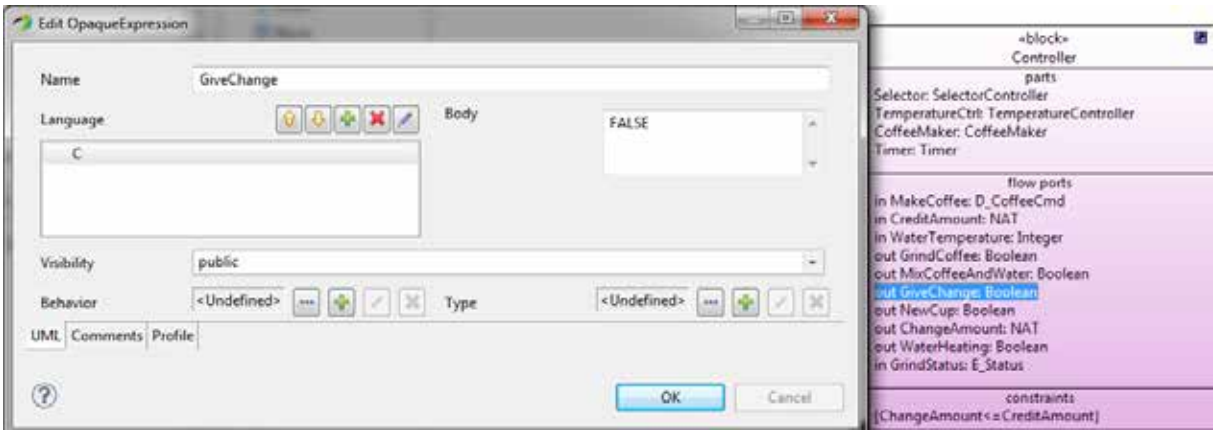

Figure 8: example of an opaque expression which assigns a default value to the output *GiveChange* which is typed as a Boolean.

RULE_30 All the flow ports, property nodes et flow properties typed with a Primitive type or an Enumeration node shall have a default value.

RULE_31 The body of the Opaque Expression shall contains only the initial value of the corresponding SysML element.

6.12 INSTANCE SPECIFICATION

An instance specification is used to instantiate a block and to initialize the block attributes (inputs, outputs, local variables, part properties) with specific values by using slots (see §6.13).

So, an instance specification of a block is translated by the SysML to B translator tool to a B module.

The Figure 9 shows an example of how instance specifications can be used in order to instantiate blocks.

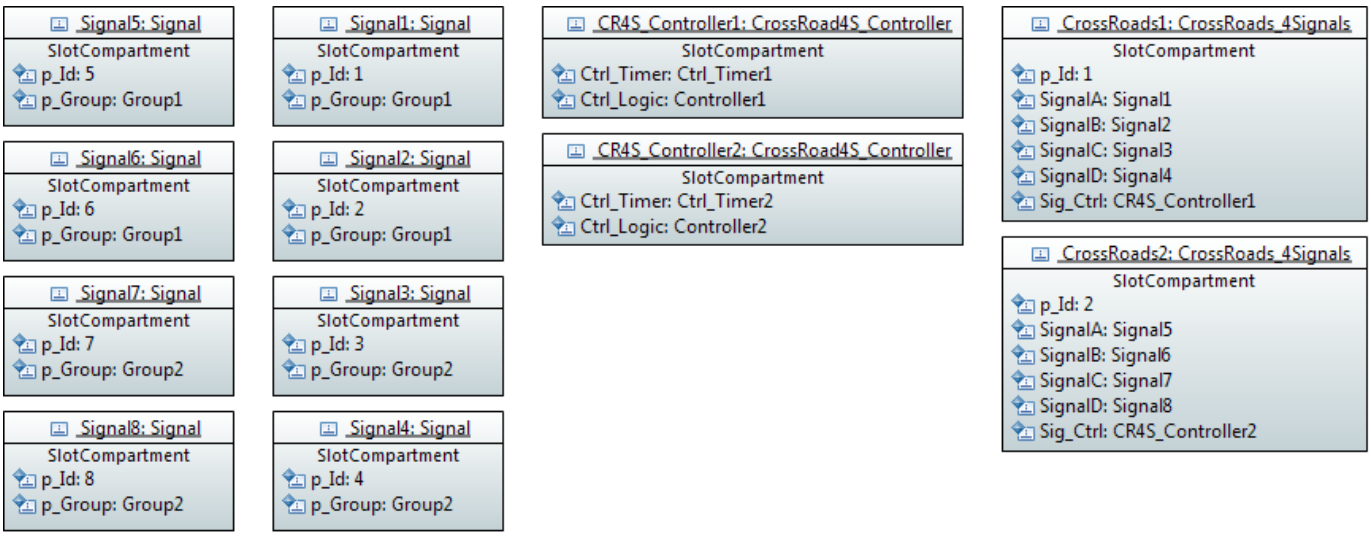

Figure 9: example of the use of instance specifications. Here, *CrossRoads1* and *CrossRoads2* are two instance specifications of the block *CrossRoads_4Signals*. These instance specifications contain slots that initialize the value of the local variable *p_Id* with an opaque expression, and slots that initialize the value of parts with instance value. Thus, the instance *CrossRoads1* has its part *SignalA* initialized with the instance specification *Signal1*.

RULE_32 An instance specification shall have its classifier parameter defined with one and only one block.

RULE_33 All the instance specifications defined for the same classifier (*i.e.* for the same block), shall have a different name.

6.13 SLOT

A slot is a member of an instance specification. It corresponds to a member of the block for which its owning Instance Specification is defined. A slot defines an initial value for this member.

So, a slot can be associated to a flow port, a property node or a part property of the block.

The Figure 9 shows how slots can be used by instance specifications.

RULE_34 The feature defined by a slot shall be a flow port or a property node typed with a primitive type or an enumerated type, or a part property of the block.

RULE_35 Each instance specification of a block shall contain a slot for each part property of the corresponding block.

RULE_36 An instance specification cannot contain several slots that define the same feature of the corresponding block.

RULE_37 A slot which define a flow port or a property node (i.e. a local variable of the block) typed with a primitive type or an enumerated type shall have a value defined with one and only one opaque expression (see §6.11).

RULE_38 A slot which define a part property shall have a value defined with one and only one Instance Value (see §**Erreur ! Source du renvoi introuvable.**).

RULE_39 An instance specification can be used at maximum by only one slot of the model which defines a part property.

6.14 INSTANCE VALUE

An Instance Value is used to define the value of a slot when this slot correspond to a block part property. The value defined by an Instance Value is an instance specification.

For example, in , the slots that corresponds to part properties are initialized with instance value.

RULE_40 A slot value shall be defined with one and only one Instance Value.

RULE_41 The instance parameter of an instance value shall be filled with an Instance Specification that has the same type than the owning slot of the Instance Value.

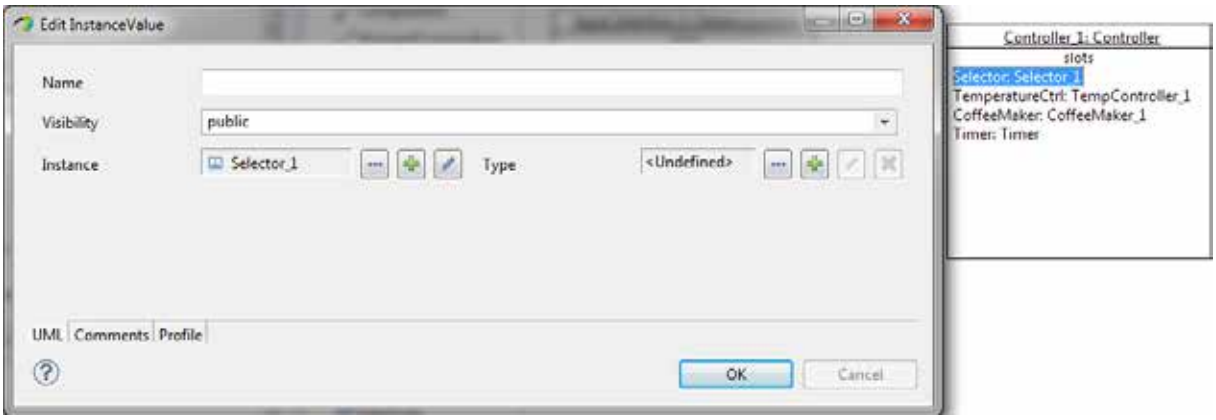

Figure 10: example of the edition of an Instance Value. The slot Selector has its value set to the instance specification *Selector_1* by using an Instance Value.

7 INTERNAL BLOCK DIAGRAM

7.1 PART PROPERTY NODE

A part property node is used by the SysML to B translator tool when it computes connections between block instances through flow ports.

A part property represents a part of a block.

RULE_42 A Part Property shall be typed with a block which is under its owning block in the hierarchical tree of the model.

RULE_43 A part property cannot send and give information to another part property (see Figure 11).

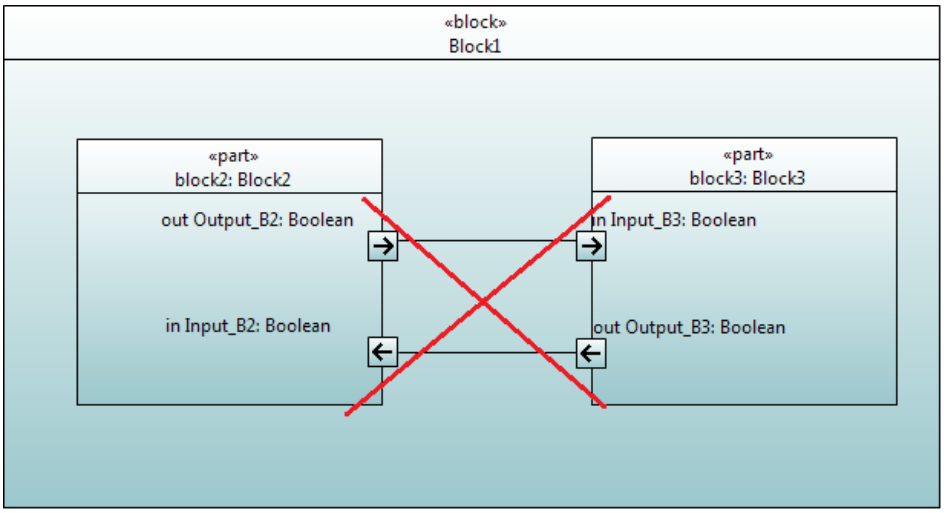

Figure 11: two parts cannot be connected in two different directions. Cycles are not allowed.

7.2 CONNECTOR PATH

The connector path is interpreted by the SysML to B translator tool as a link between flow ports. Flow ports linked with a connector path can exchange data. Thus, a flow port receives data from its previous flow port and sends data to it next flow port(s).

RULE_44 A connector path can only be connected to flow ports.

RULE_45 A connector path shall connect flow ports with a compatible direction (see Figure 12 and Figure 13 as example).

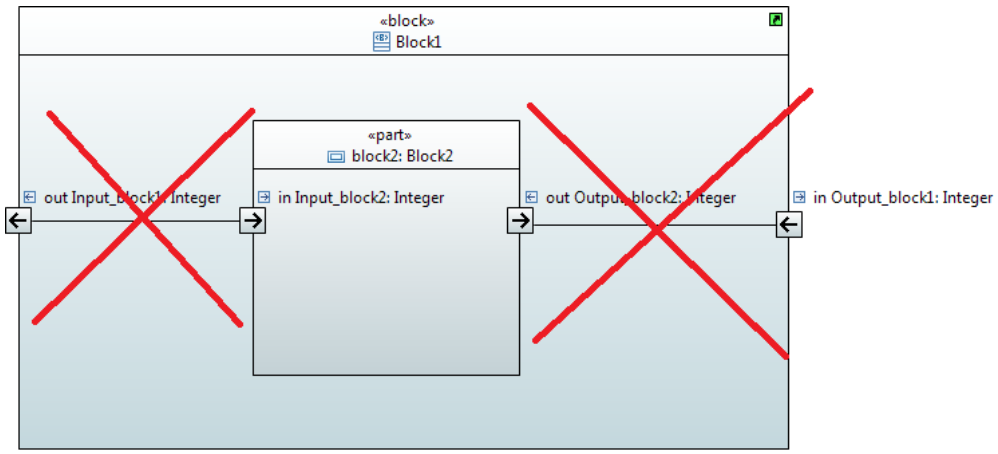

Figure 12: example of a wrong use of connector path

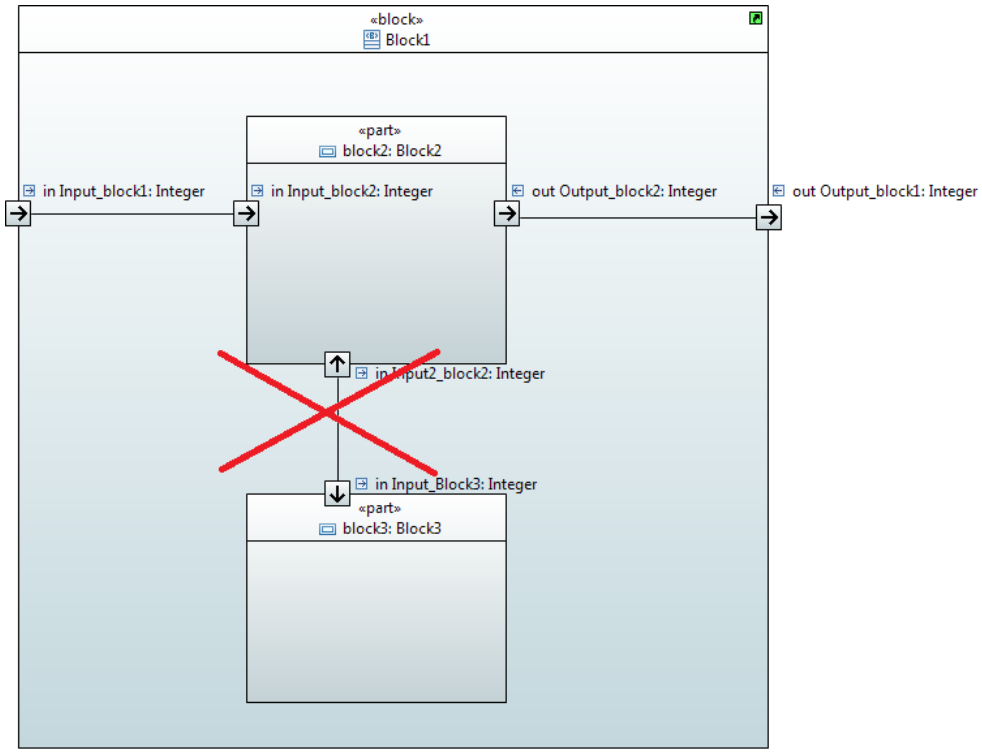

Figure 13: example of a wrong use of connector path

7.3 ATOMIC FLOW PORT NODE

A flow port with "In" direction is seen as an input of its owning block and a flow port with "out" direction is seen as an output of its owning block.

We call the previous flow port of a selected flow port, the flow port that is connected by a connector path and gives the information to the selected flow port. We call the next flow port(s) of a selected flow port, the flow port(s) that is(are) connected by a connector path and receives the information from the selected flow port. For example, in the example depicted in the Figure 14, the flow port "Input_block2" has "input_block1" as previous flow port and has no next flow port. The flow port "Output_block2" has "Output_block1" as next flow port and has no previous flow port.

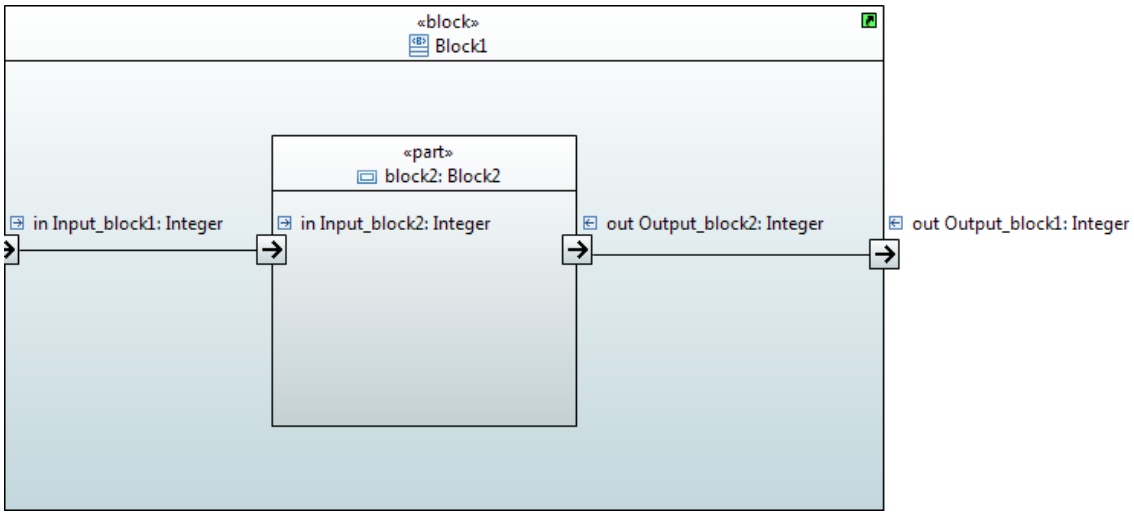

Figure 14: example of final and non-final flow ports

RULE_46 Flow ports can only be owned by blocks.

RULE_47 A flow port shall have a direction specified with the value "In" or "Out". Value "InOut" is not allowed.

RULE_48 A flow port shall be typed with a primitive type node, with a Enumeration node, with a DataType node or with a flow specification node.

RULE_49 The type of a flow port shall be identical to the type of its connected flow ports.

RULE_50 A flow port cannot be connected to more than one previous flow port (see Figure 15).

RULE_51 A flow port with the "In" direction shall be connected to one and only one other previous flow port.

RULE_52 A flow port can be connected to other flow port(s) only with a connector path.

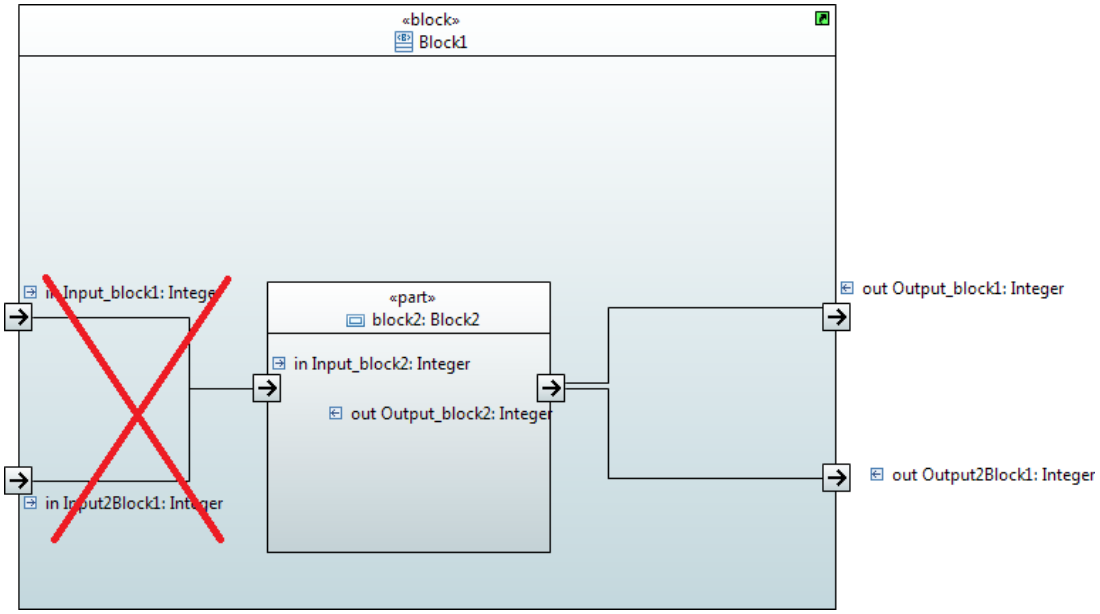

Figure 15: example of possible connections between flow ports

7.4 OPAQUE EXPRESSION (FOR DEFAULT VALUE)

See §6.11.

8 OTHER DIAGRAMS

State machine diagrams, Requirement diagrams, Sequence diagrams and corresponding SysML elements are not translated by the SysML to B translator tool.