VMTS

Visual Modeling and Transformation System

T. Levendovszky, L. Lengyel, G. Mezei, H. Charaf, A Systematic Approach to Metamodeling Environments and Model Transformation Systems in VMTS, Electronic Notes in Theoretical Computer Science 127 (1) (2005) 65–75.

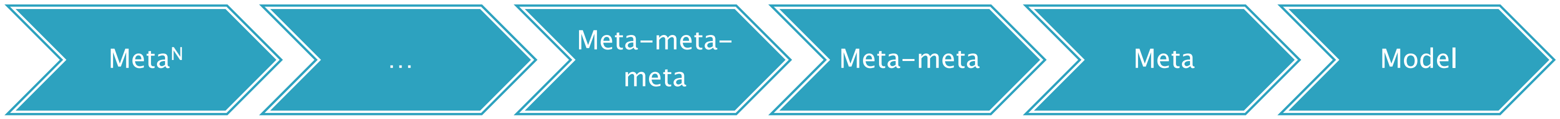
Dylan Kiss
University of Antwerp
dylan.kiss@student.uantwerpen.be

Metamodeling environment

- N-layer metamodeling environment
- Simplified UML class diagrams
- UML class diagram instantiation:
  - UML object diagram
  - UML class diagram
  - Metamodel of UML class diagram
- Two more layers:
  - Read-only meta-metamodel
  - Internal structure: labeled directed graph

Model storage

- AGSI
  - Attributed Graph Architecture Supporting Inheritance
- Every model can be a metamodel for others

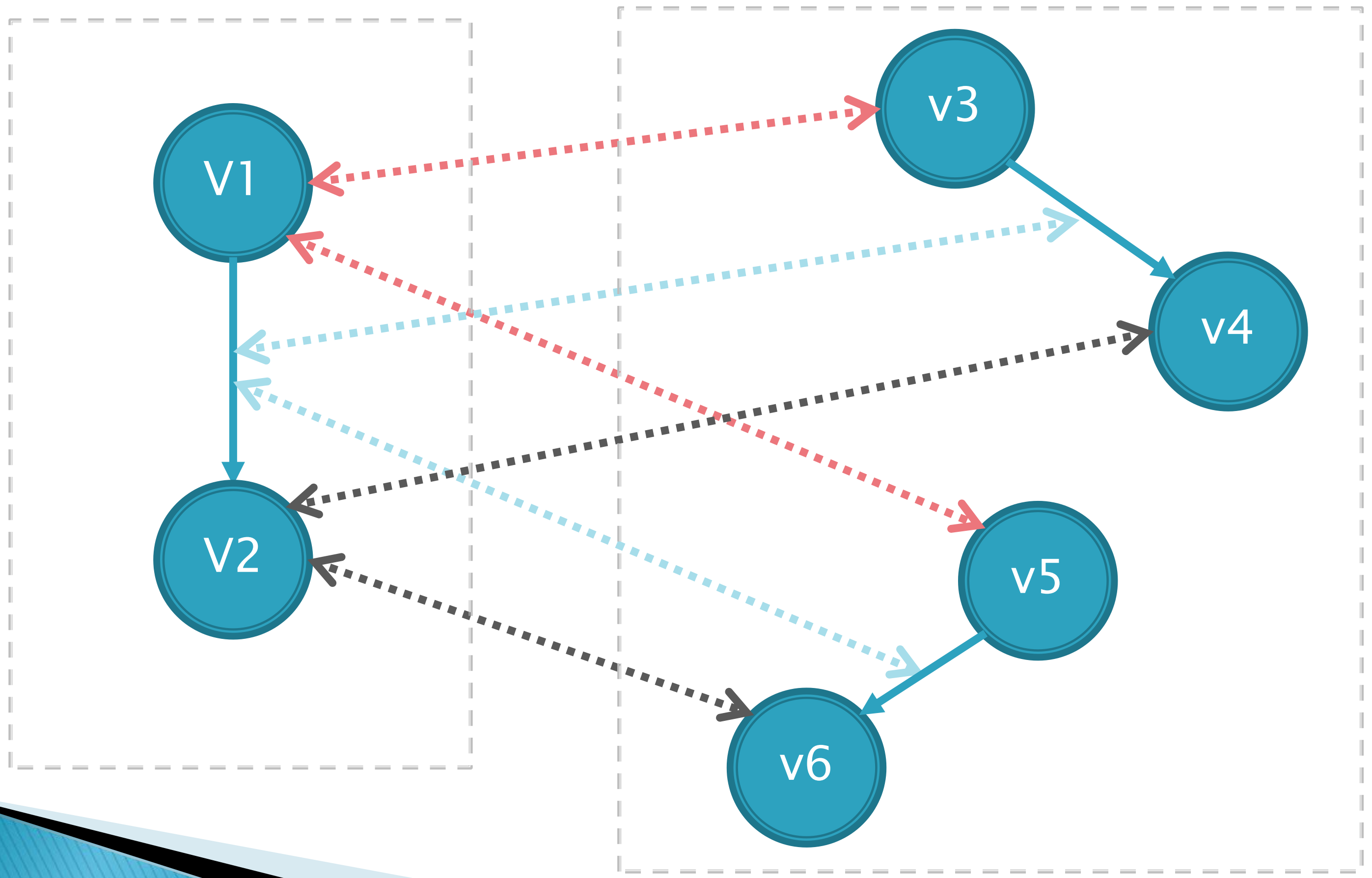


AGSI

- 3 basic graph constructs:
  - Nodes
  - Directed edges
  - Labels
- Metamodeling needs extra things:
  - Type-instance mapping
  - Containment
  - Inheritance
  - Association classes

AGSI

- Type-instance mapping



AGSI

- Containment
  - Parent–child bidirectional mapping
- Inheritance
  - Directed mapping from descendants to ancestors
- Association classes
  - Pseudo–nodes

AGSI

- Model attributes (labels in directed graph)
  - Stored in XMI-like format
  - Meta-attributes that can be instantiated are stored in XSD file
    - Schema for XML file on instance level

Model transformations

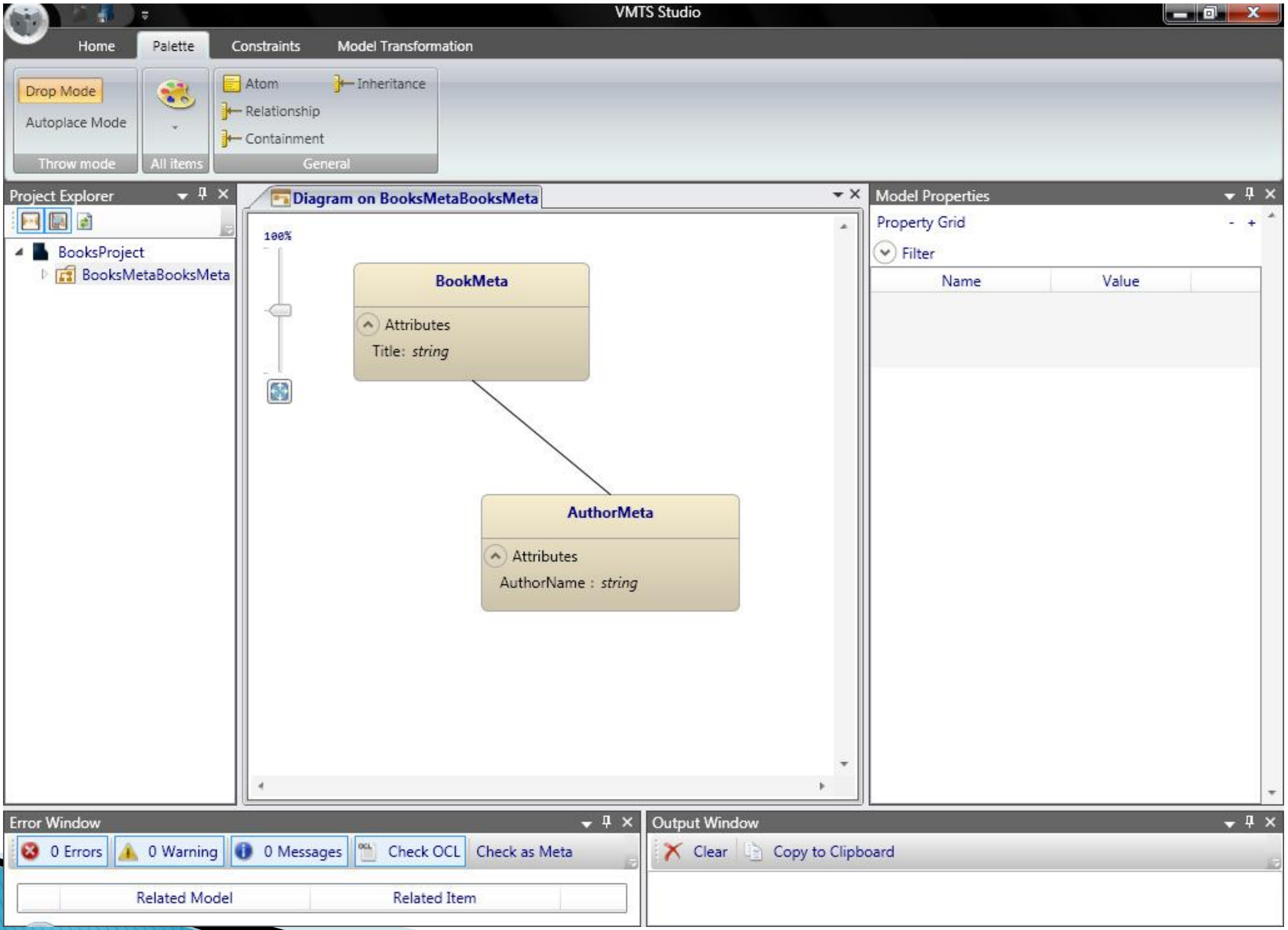
- Traversing Model Processors
  - Create node
  - Connect nodes
  - Delete node
  - Delete edge
  - Set label
- Regular objects in OO programming language

Model transformation

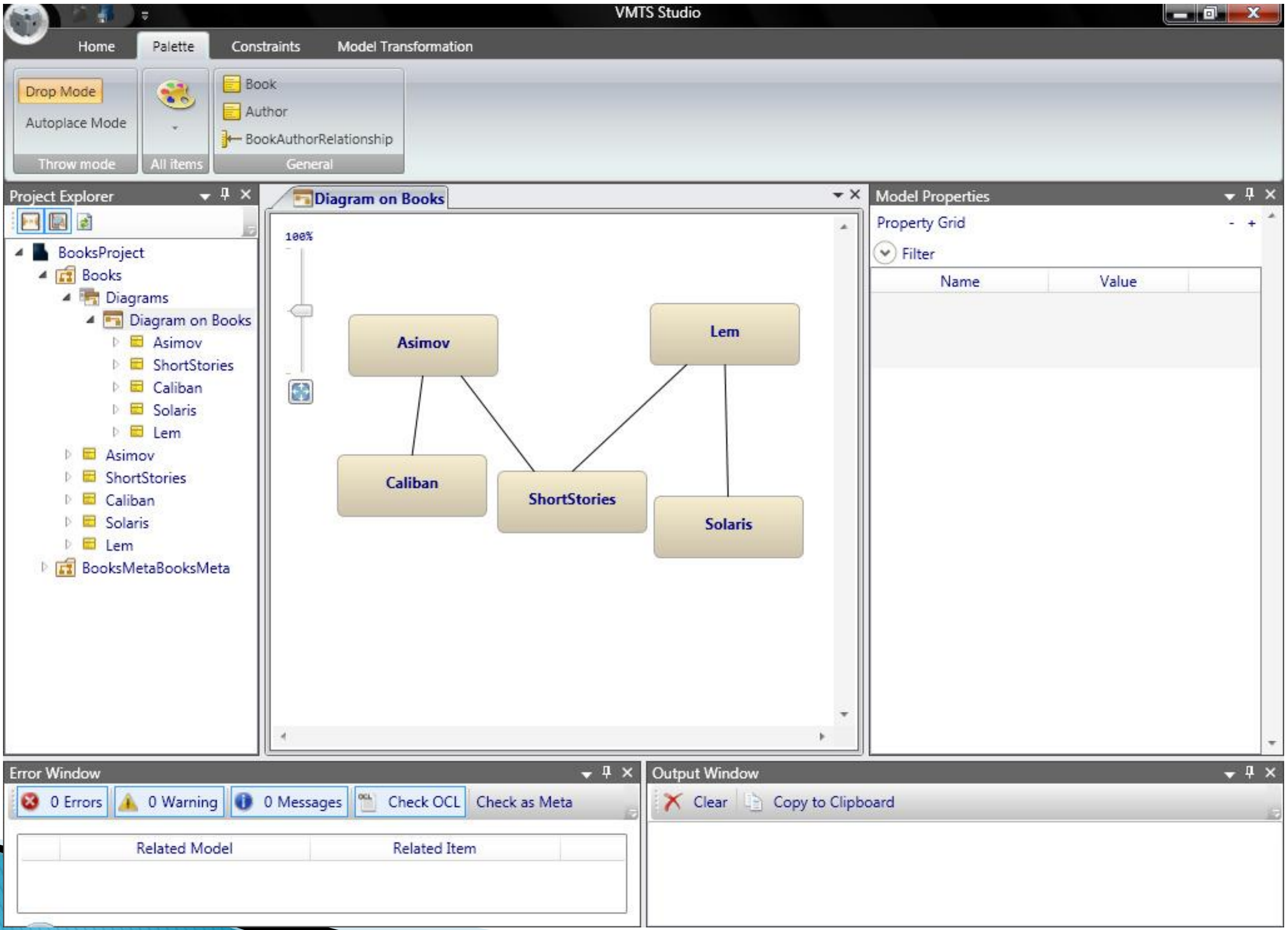
- Visual Model Processors
  - Graph rewriting
  - Rules with LHS and RHS

- Rules specified in terms of metamodel
- Attribute transformation with XSLT scripts

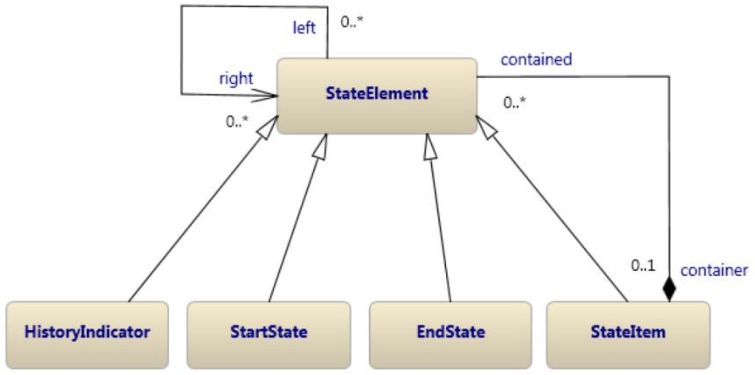
Modeling interface

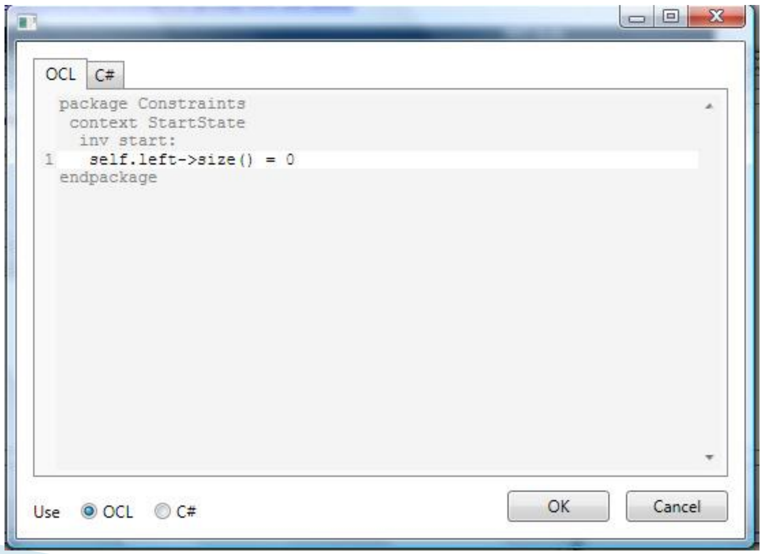


Modeling interface



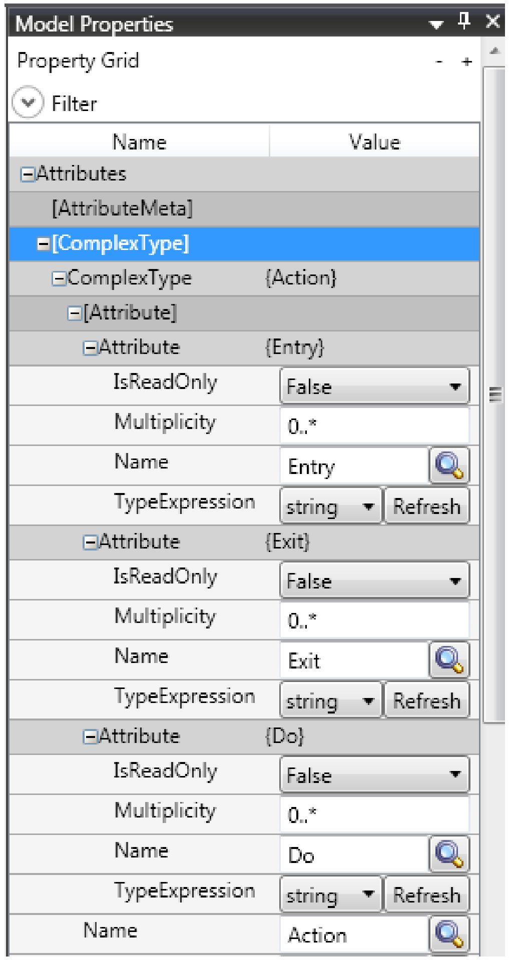
Modeling interface





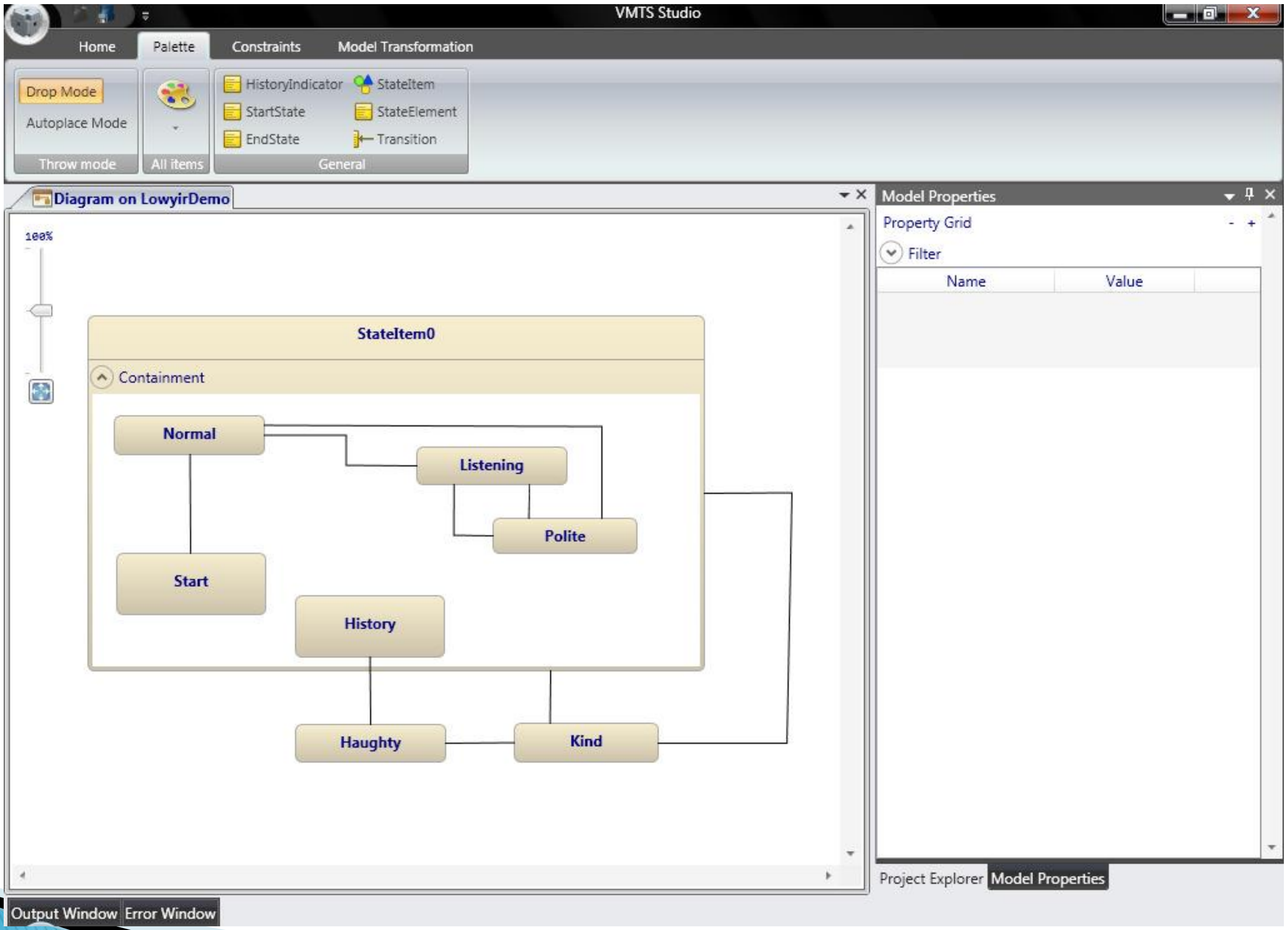
```
package Constraints
 context StartState
  inv start:
   self.left->size() = 0
endpackage
```

Use [✓ OCL] C#

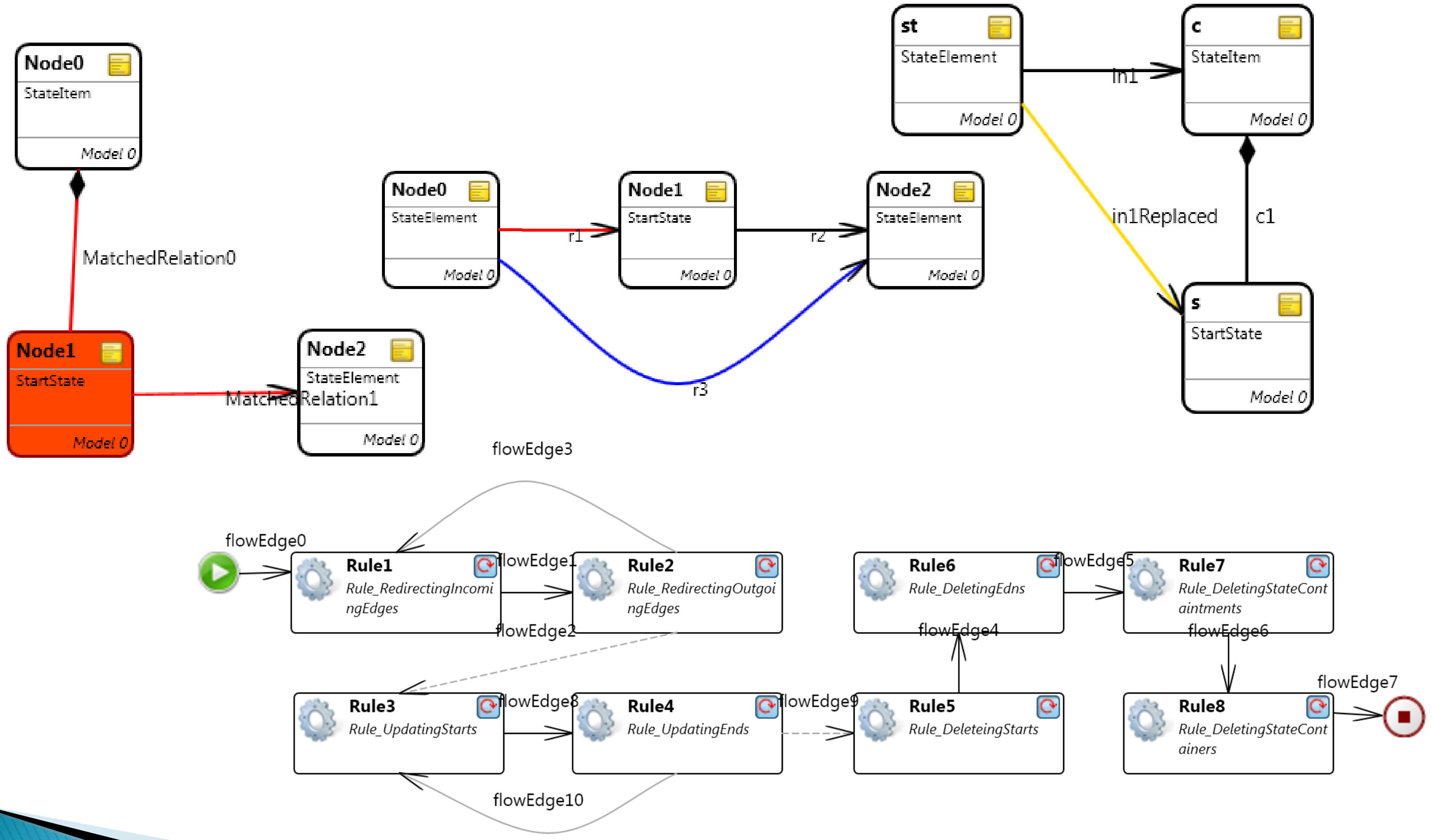


Name	Value
⊟Attributes	
[AttributeMeta]	
⊟[ComplexType]	
⊟ComplexType	{Action}
⊟[Attribute]	
⊟Attribute	{Entry}
IsReadOnly	False
Multiplicity	0..*
Name	Entry
TypeExpression	string Refresh
⊟Attribute	{Exit}
IsReadOnly	False
Multiplicity	0..*
Name	Exit
TypeExpression	string Refresh
⊟Attribute	{Do}
IsReadOnly	False
Multiplicity	0..*
Name	Do
TypeExpression	string Refresh
Name	Action

Modeling interface



Transformation interface



Planned work

- Role-Playing Game modeling in VMTS
  - Metamodel (abstract syntax)
  - Concrete visual syntax
  - Transformation rules
    - Operational semantics
    - Denotational semantics
  - Compare with AToMPM and state advantages and disadvantages

Questions?