

# 25.3.1. The Metamodeling Architecture of MetaCASE Tool MOFLON



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

**From: 10 Jahre Dresden-OCL – Workshop**  
<http://dresden-ocl.sourceforge.net/>  
<http://dresden-ocl.sourceforge.net/10years.html>



ES Real-Time Systems Lab

Prof. Dr. rer. nat. Andy Schürr

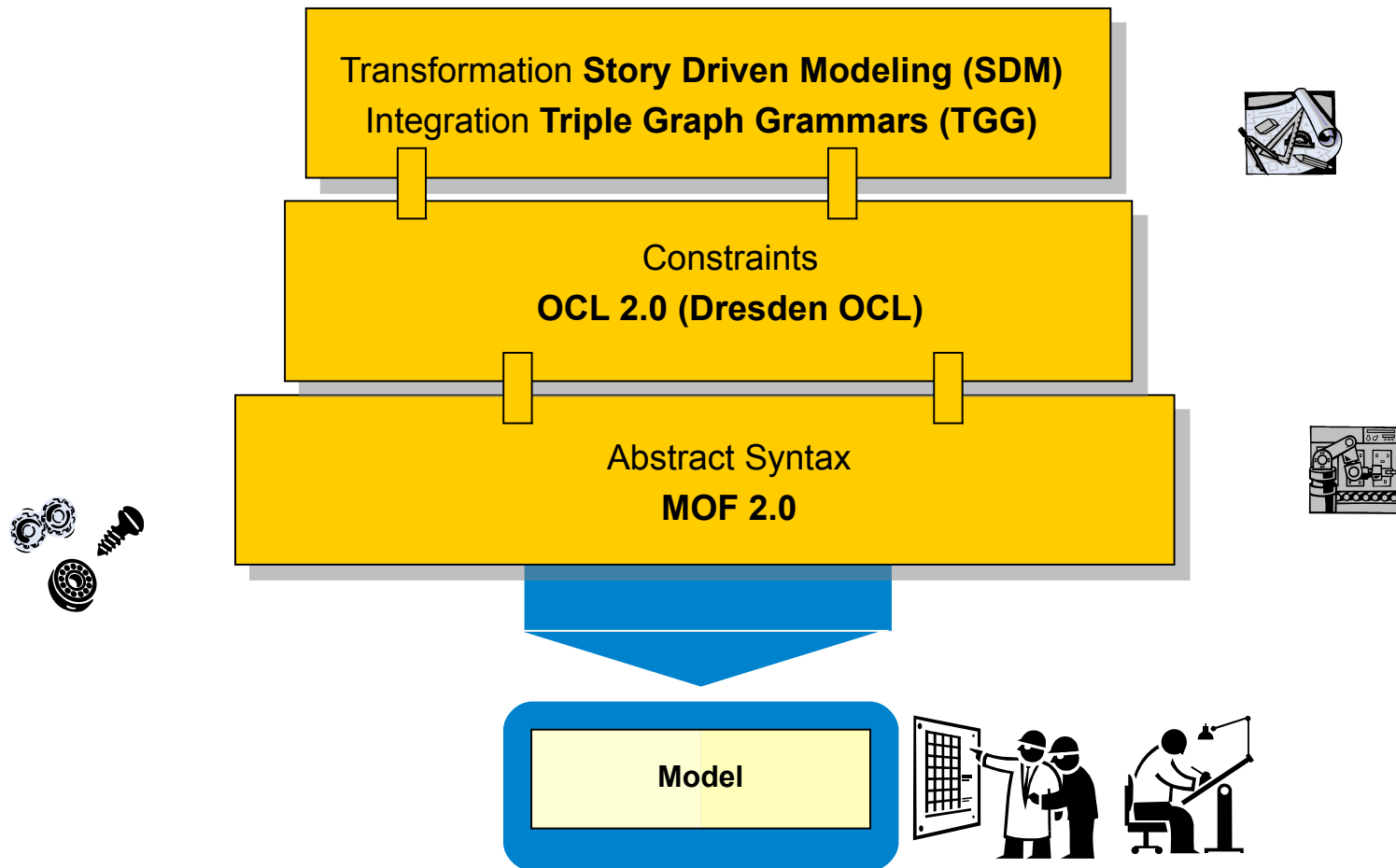
Dept. of Electrical Engineering and Information Technology  
Dept. of Computer Science (adjunct Professor)

**Felix Klar**

Felix.Klar@es.tu-darmstadt.de

[www.es.tu-darmstadt.de](http://www.es.tu-darmstadt.de)

# Metamodel Architecture of MOFLON



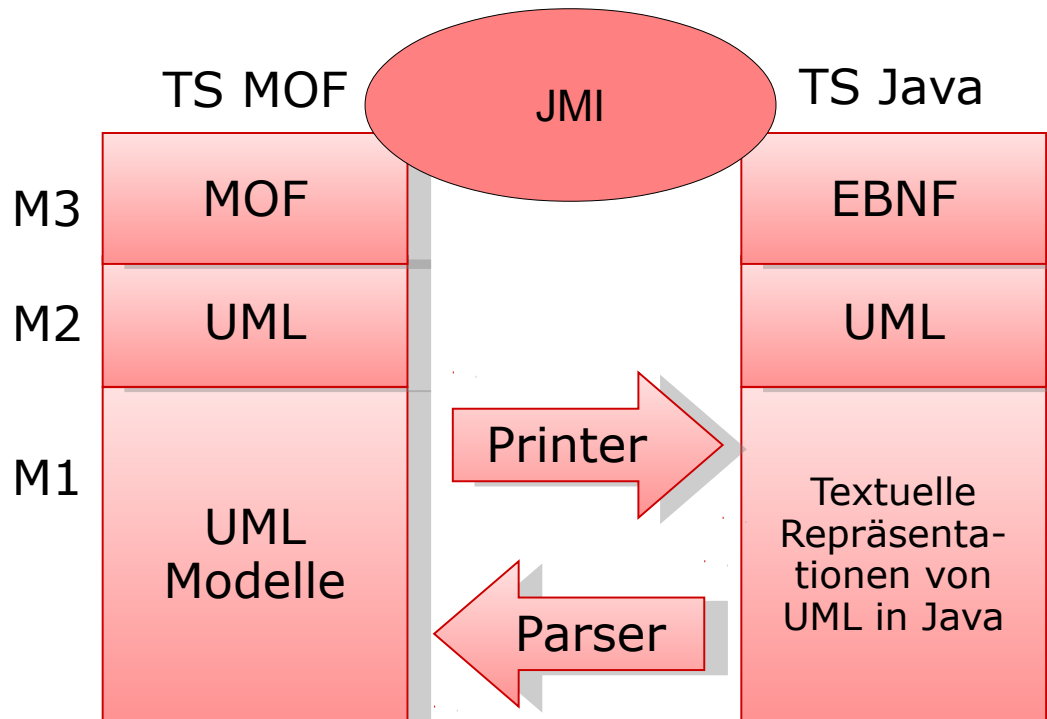
# MOFLON MetaCASE – Main Features

- MOF2.0 editor (draw metamodels that comply to MOF2.0 standard)
  - build Domain Specific Languages (DSLs)
    - based on the CASE-tool framework Fujaba
    - possibility to extend MOFLON by own plugins
- interoperability (import / export)
- transform metamodel instances with model transformations (SDM, TGG)
- generate code (JMI-compliant) from DSLs
- instantiate models of the DSL (= repositories)
- basic editing support for generated repositories



# Einschub: JMI: Transformative TS-Brücke für MOF und Java, Sprache UML

Java Metadata Interchange (JMI) ist eine TS-Halb-Brücke für MOF und EBNF-Space, für die Sprache UML



# (OCL) Constraints in MOFLON – MOF Editor

- MOF allows to add constraints to every MOF element
- MOFLON has an underlying MOF metamodel repository  
→ MOFLON MOF editor may add constraints to elements

The screenshot displays the MOFLON MOF Editor interface. A dialog box titled "Edit MOF Constraint" is open, showing the "General" tab. The "Name" field contains "attrNamesMustDiffer", the "Language" is set to "OCL", and the "Body" contains the OCL expression: `inv:attrs->forall(a1,a2:Attribute|a1<>a2 implies a1.name <> a2.name)`. The "Language" dropdown is highlighted with a red dashed circle. Below the dialog, a red dashed arrow points to a question mark icon in the bottom toolbar, with the text "validate constraints" written below it. The background shows a class diagram with classes "Association", "Clazz", and "Attribute". The "Clazz" class is circled in red, and its "attrs" attribute is also circled. The "Attribute" class is also circled in red. The diagram shows relationships between these classes, including "AssocToSource", "AssocToTarget", "AttrToType", "ClassToAttrs", "ClassToClass", and "children".





```
619
620 public Collection<String> refConstraintNames() {
621     Collection<String> constraintNames = new java.util.HashSet<String>();
622
623     constraintNames.add("attrNamesMustDiffer");
624
625     return constraintNames;
626 }
627
628 public javax.jmi.reflect.JmiException refVerifyConstraint(String constraintName) {
629     if ("attrNamesMustDiffer".equals(constraintName)) {
630         if (!evaluate_attrNamesMustDiffer()) {
631             String constraintBody = "unknown body";
632             constraintBody = "inv:attrs->forall(a1,a2:Attribute|a1<>a2 implies a1.name <> a2.name)";
633             informListener(new ConstraintEvent(this, ConstraintEvent.EVENT_OCL_INVARIANT, "constraintName", false));
634
635             return new javax.jmi.reflect.ConstraintViolationException(
636                 constraintBody, this, "constraint named '" + constraintName + "' is violated in instance: " + this);
637         } else {
638             informListener(new ConstraintEvent(this, ConstraintEvent.EVENT_OCL_INVARIANT, "constraintName", true));
639         }
640     }
641     return null;
642 }
643
644 public Collection<javax.jmi.reflect.JmiException> refVerifyConstraints(boolean deepVerify) {
645     Collection<javax.jmi.reflect.JmiException> invalidConstraints = new org.moflon.collections.implementation.JmiSetImpl<
646
647     for (String constraintName : refConstraintNames()) {
648         javax.jmi.reflect.JmiException constraintException = refVerifyConstraint(constraintName);
649
650         if (constraintException != null) {
651             invalidConstraints.add(constraintException);
652         }
653     }
654
655     if (deepVerify) {
656     }
657
658     if (invalidConstraints.size() > 0) {
659         return invalidConstraints;
660     } else {
661         return null;
662     }
663 }
664
```

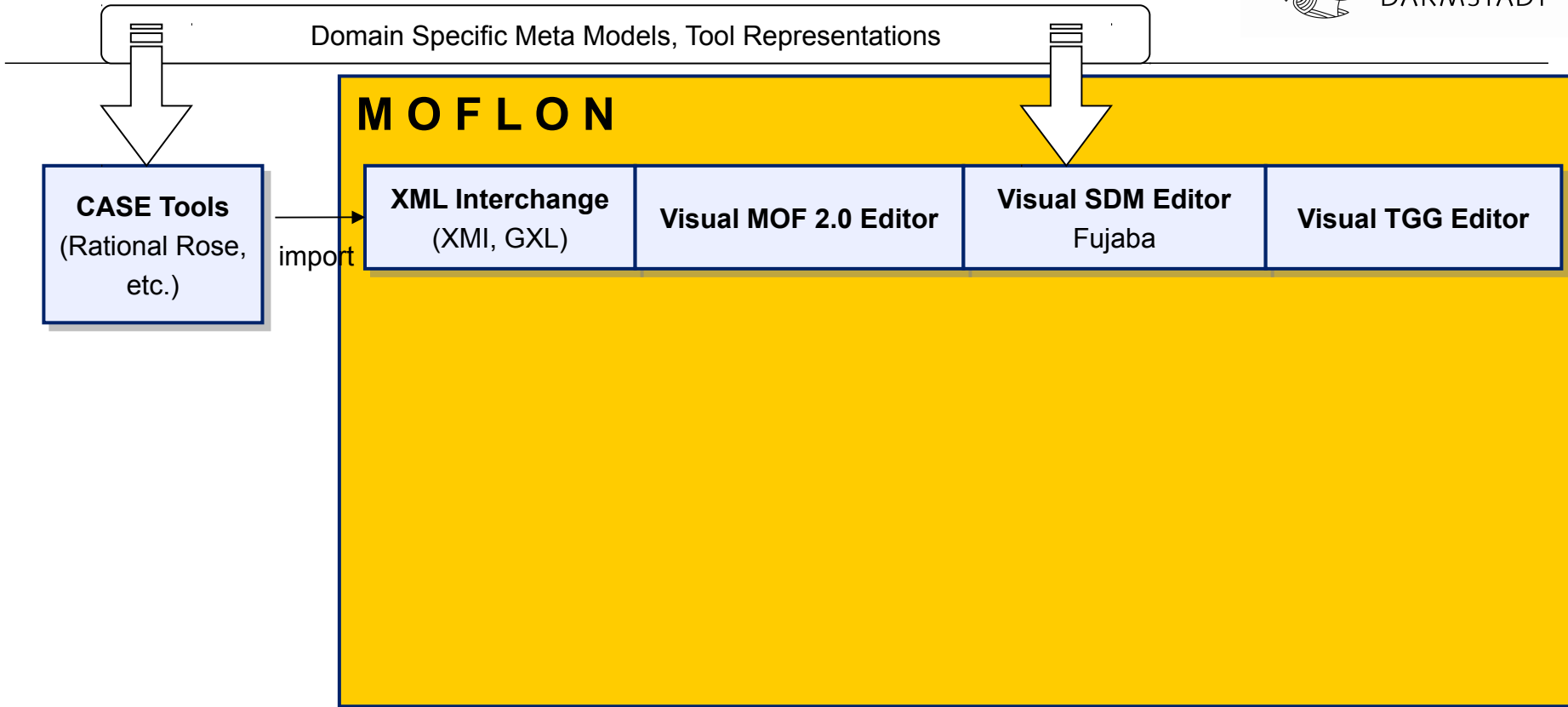


```
ClazzImpl.java X
348 // generating constraint evaluation method attrNamesMustDiffer
349 public boolean evaluate_attrNamesMustDiffer() {
350     // Variables
351     final tudresden.oc120.core.lib.JmiOclFactory tudOcl20Fact0 = tudresden.oc120.core.lib.JmiOclFactory.getInstance(refOutermostPackage());
352     final tudresden.oc120.core.lib.OclCollectionType tudOcl20Type1 = tudOcl20Fact0.getOclModelTypeFor("cd_metamodel::Attribute").getOclBagType();
353     final tudresden.oc120.core.lib.OclPrimitiveType tudOcl20Type2 = tudresden.oc120.core.lib.OclPrimitiveType.getOclString();
354     final tudresden.oc120.core.lib.OclModelType tudOcl20Type0 = tudOcl20Fact0.getOclModelTypeFor("cd_metamodel::Clazz");
355
356     // Invariant
357     final tudresden.oc120.core.lib.OclModelObject tudOcl20Var0 = (tudresden.oc120.core.lib.OclModelObject) tudOcl20Fact0.getOclRepresentationFor(
358         tudOcl20Type0, this);
359     final tudresden.oc120.core.lib.OclBag tudOcl20Exp0 = tudresden.oc120.core.lib.Ocl.toOclBag(tudOcl20Var0.getFeature(tudOcl20Type1, "attrs"));
360     final tudresden.oc120.core.lib.OclIterator tudOcl20Iter0 = tudOcl20Exp0.getIterator();
361     final tudresden.oc120.core.lib.OclBooleanEvaluatable tudOcl20Eval0 = new tudresden.oc120.core.lib.OclBooleanEvaluatable() {
362         public tudresden.oc120.core.lib.OclBoolean evaluate() {
363             final tudresden.oc120.core.lib.OclModelObject tudOcl20Var1 = tudresden.oc120.core.lib.Ocl.toOclModelObject(tudOcl20Iter0.getValue());
364             final tudresden.oc120.core.lib.OclIterator tudOcl20Iter1 = tudOcl20Exp0.getIterator();
365             final tudresden.oc120.core.lib.OclBooleanEvaluatable tudOcl20Eval1 = new tudresden.oc120.core.lib.OclBooleanEvaluatable() {
366                 public tudresden.oc120.core.lib.OclBoolean evaluate() {
367                     final tudresden.oc120.core.lib.OclModelObject tudOcl20Var2 = tudresden.oc120.core.lib.Ocl
368                         .toOclModelObject(tudOcl20Iter1.getValue());
369
370                     //TODO: Check if VariableId is correct
371                     final tudresden.oc120.core.lib.OclBoolean tudOcl20Exp1 = tudOcl20Var2.isNotEqualTo(tudOcl20Var1);
372                     final tudresden.oc120.core.lib.OclString tudOcl20Exp2 = tudresden.oc120.core.lib.Ocl.toOclString(
373                         tudOcl20Var2.getFeature(tudOcl20Type2, "name"));
374                     final tudresden.oc120.core.lib.OclString tudOcl20Exp3 = tudresden.oc120.core.lib.Ocl.toOclString(
375                         tudOcl20Var1.getFeature(tudOcl20Type2, "name"));
376                     final tudresden.oc120.core.lib.OclBoolean tudOcl20Exp4 = tudOcl20Exp2.isNotEqualTo(tudOcl20Exp3);
377                     final tudresden.oc120.core.lib.OclBoolean tudOcl20Exp5 = tudOcl20Exp1.implies(tudOcl20Exp4);
378
379                     return tudOcl20Exp5;
380                 }
381             };
382
383             final tudresden.oc120.core.lib.OclBoolean tudOcl20Exp6 = (tudresden.oc120.core.lib.OclBoolean) tudOcl20Exp0.forAll(
384                 tudOcl20Iter1, tudOcl20Eval1);
385
386             return tudOcl20Exp6;
387         }
388     };
389
390     final tudresden.oc120.core.lib.OclBoolean tudOcl20Exp7 = (tudresden.oc120.core.lib.OclBoolean) tudOcl20Exp0.forAll(tudOcl20Iter0, tudOcl20Eval0);
391
392     return tudOcl20Exp7.isTrue();
393 }
```

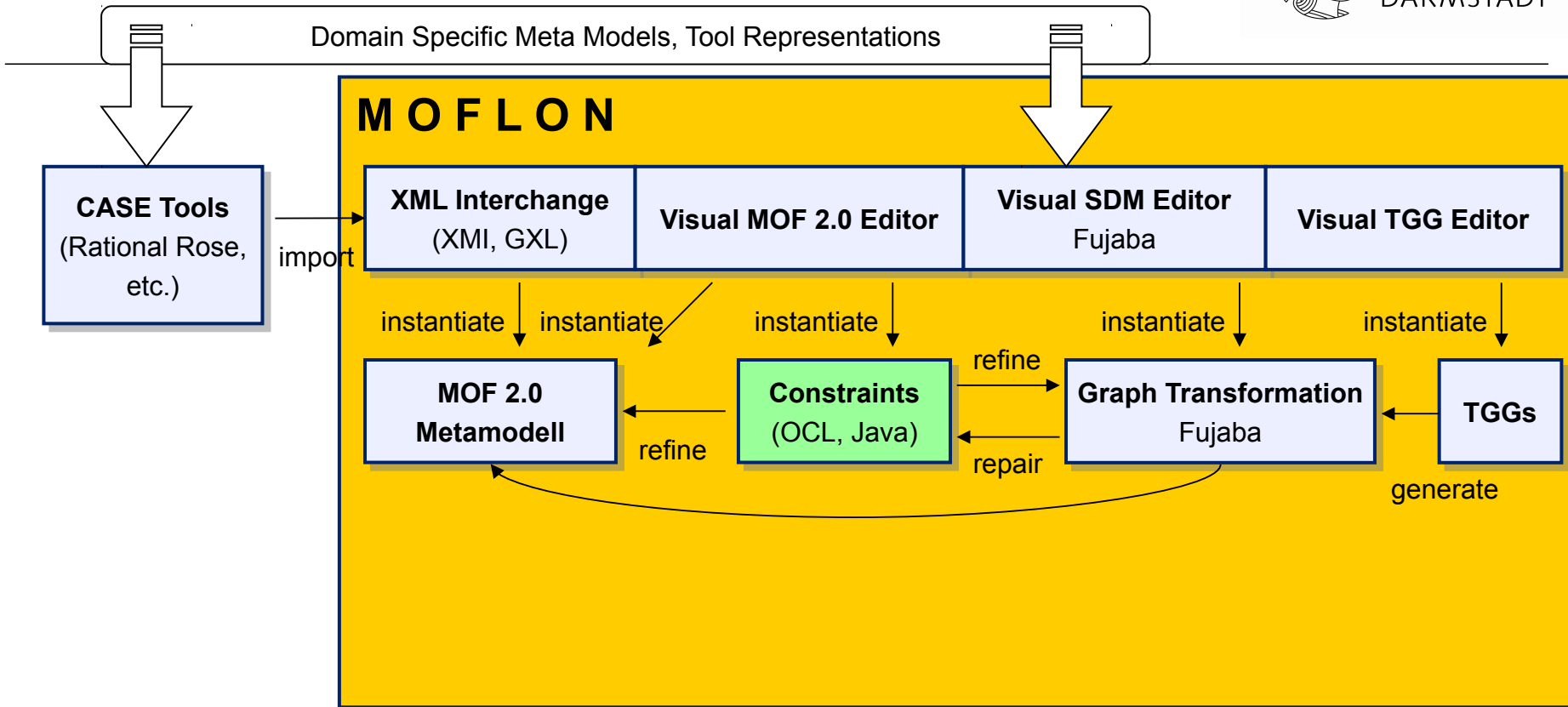




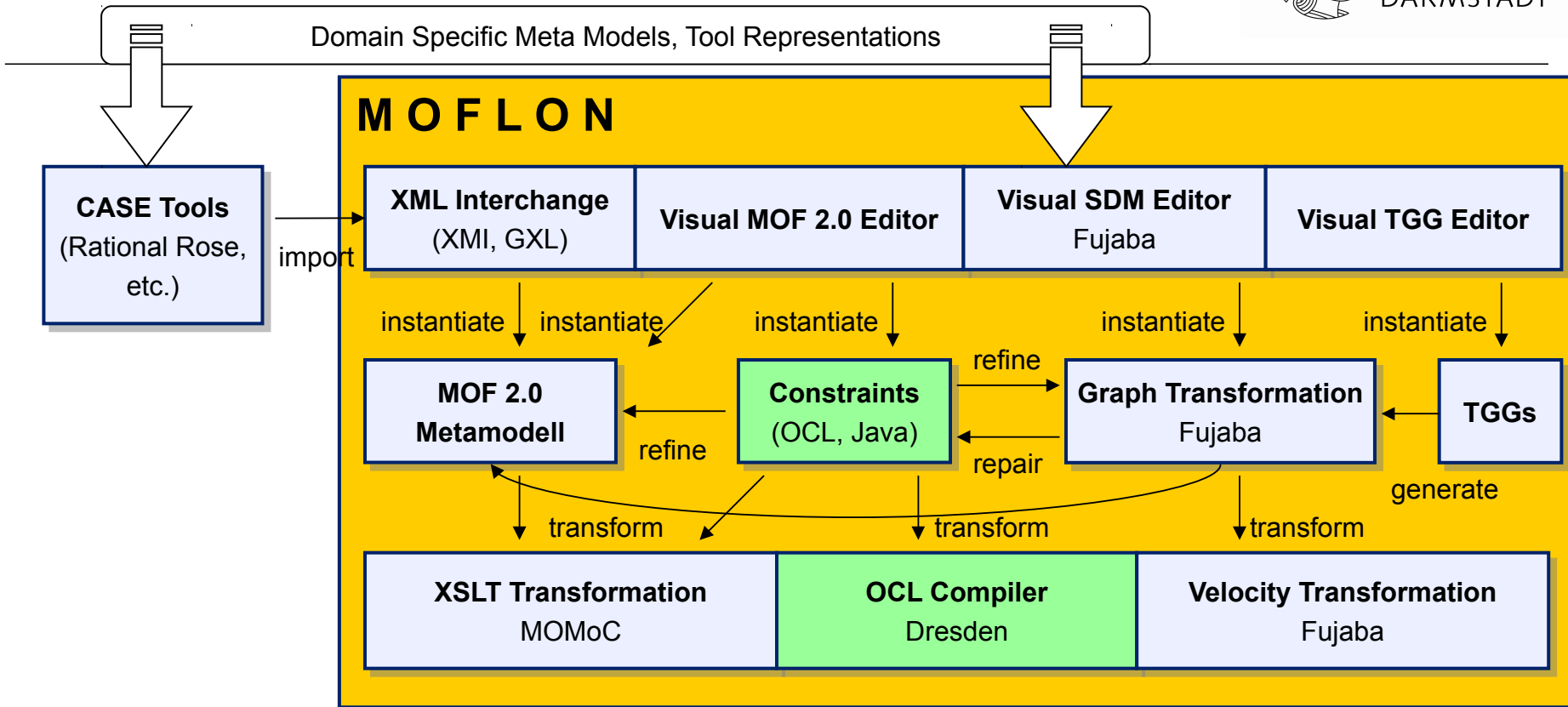
# MOFLON – Architecture



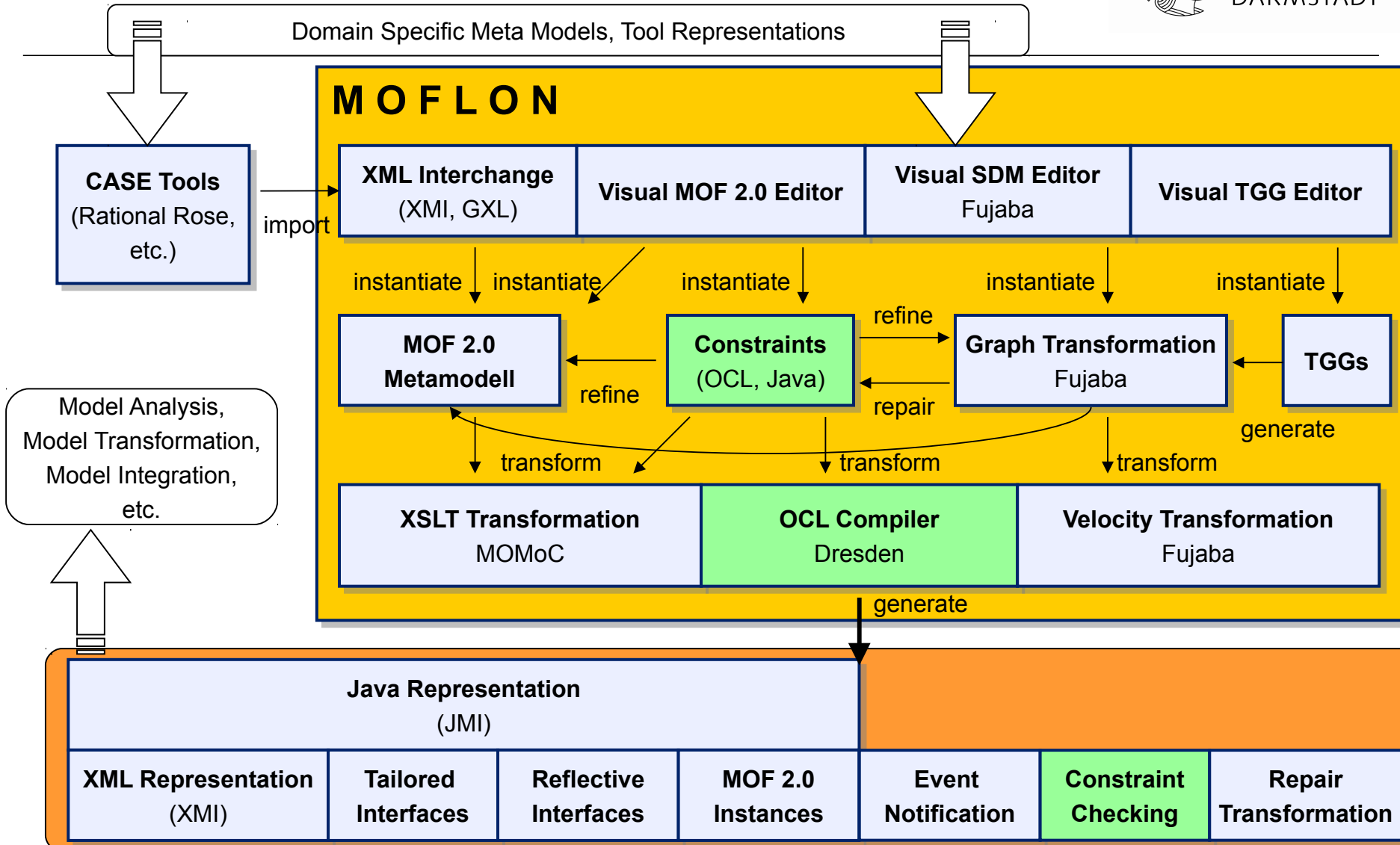
# MOFLON – Architecture



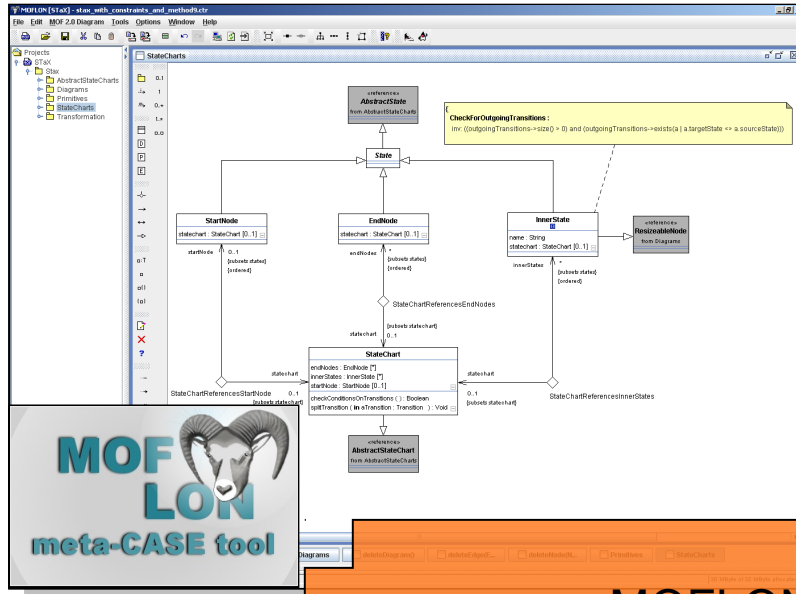
# MOFLON – Architecture



# MOFLON – Architecture

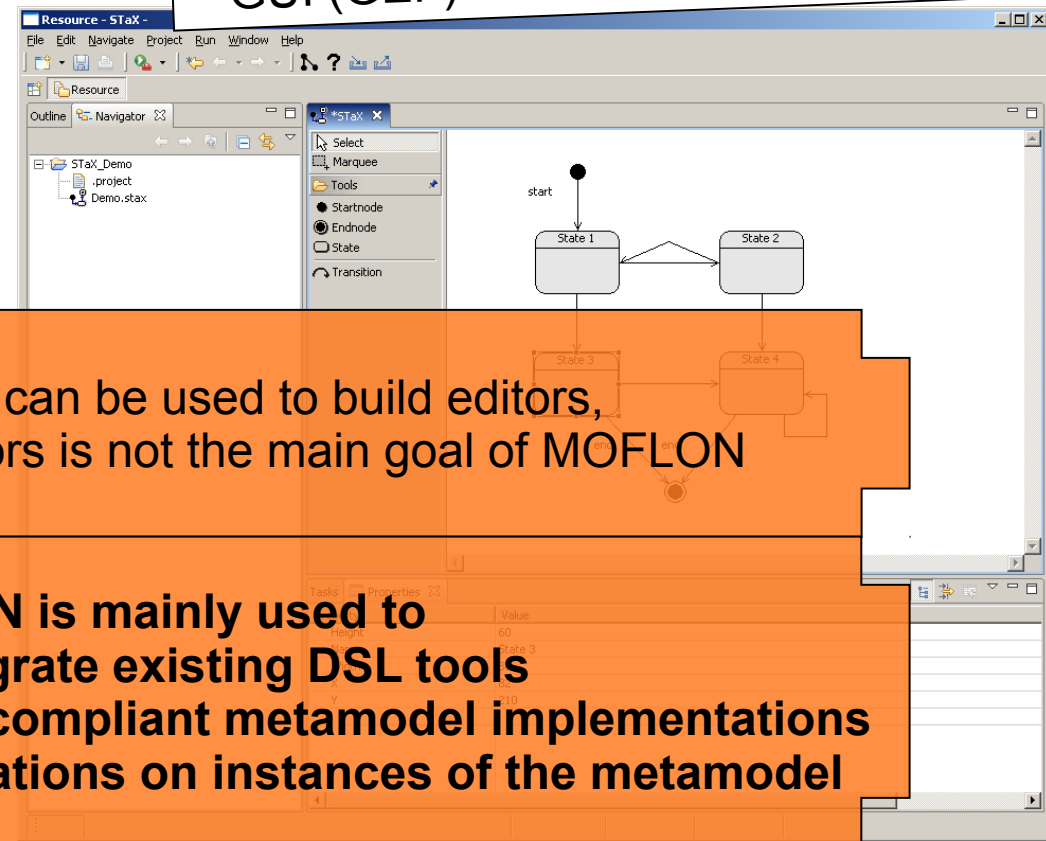


# 26.2 MOFLON Case Study – Statechart Editor (STaX)

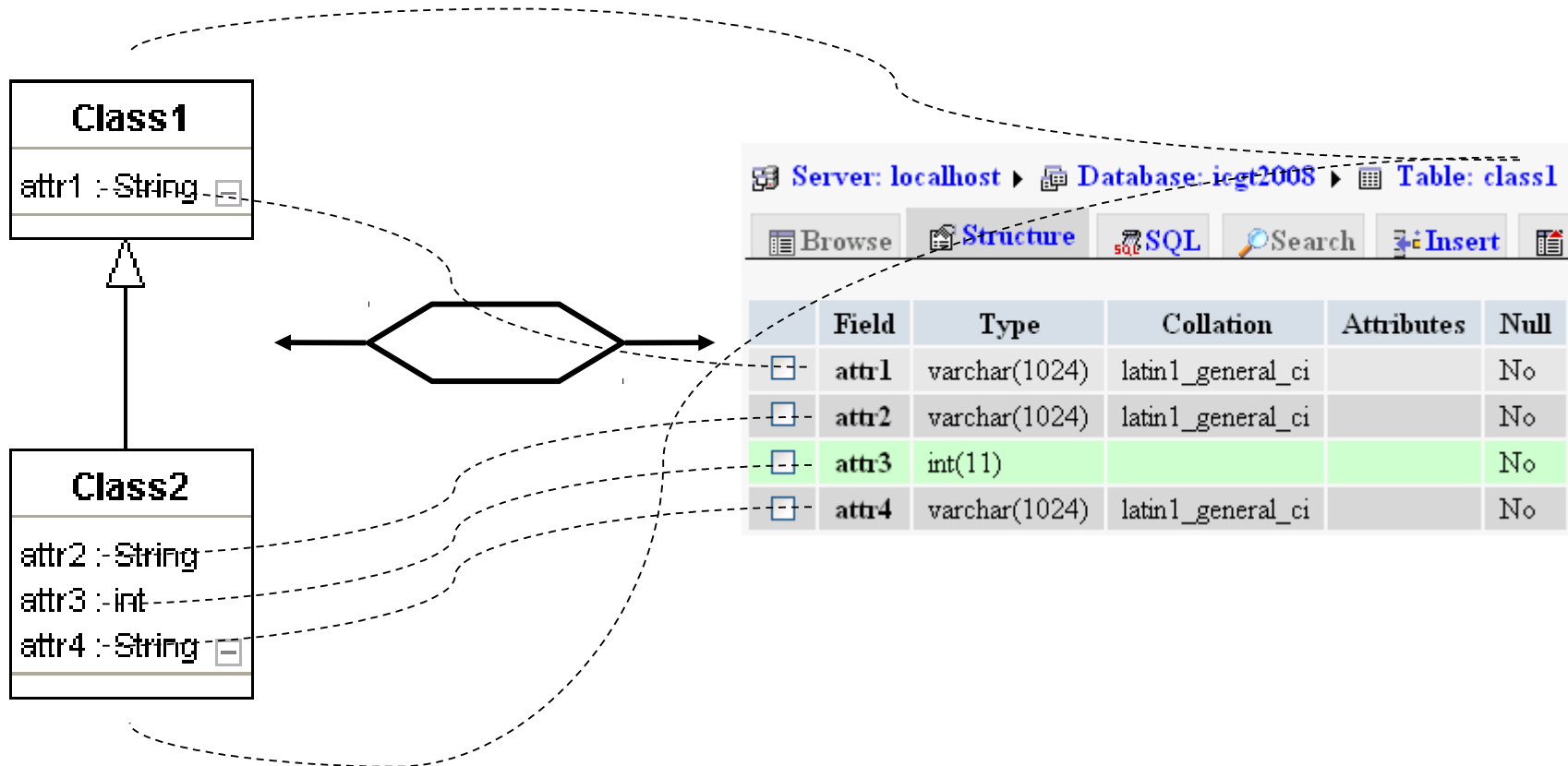


Editor:

- data structure (MOFLON repository)
- GUI (GEF)



# Integration Example with TGG – Class diagrams / database schemata



domain specific language,  
e.g. Class Diagrams

domain specific language,  
e.g. Database Schemata

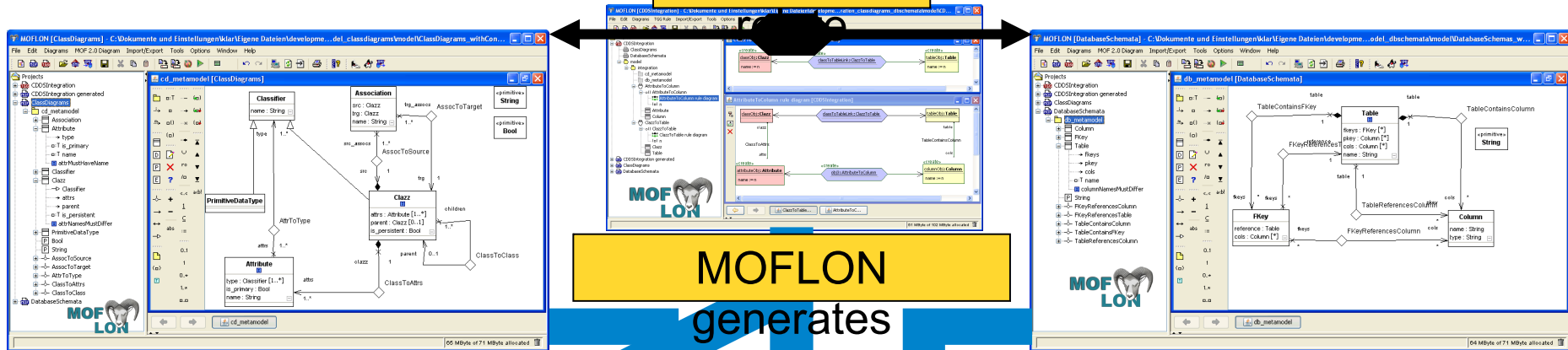
# Case Study 2: Tool Integration Scenario

## TiECDDS: (ClassD / DatabaseSchema)

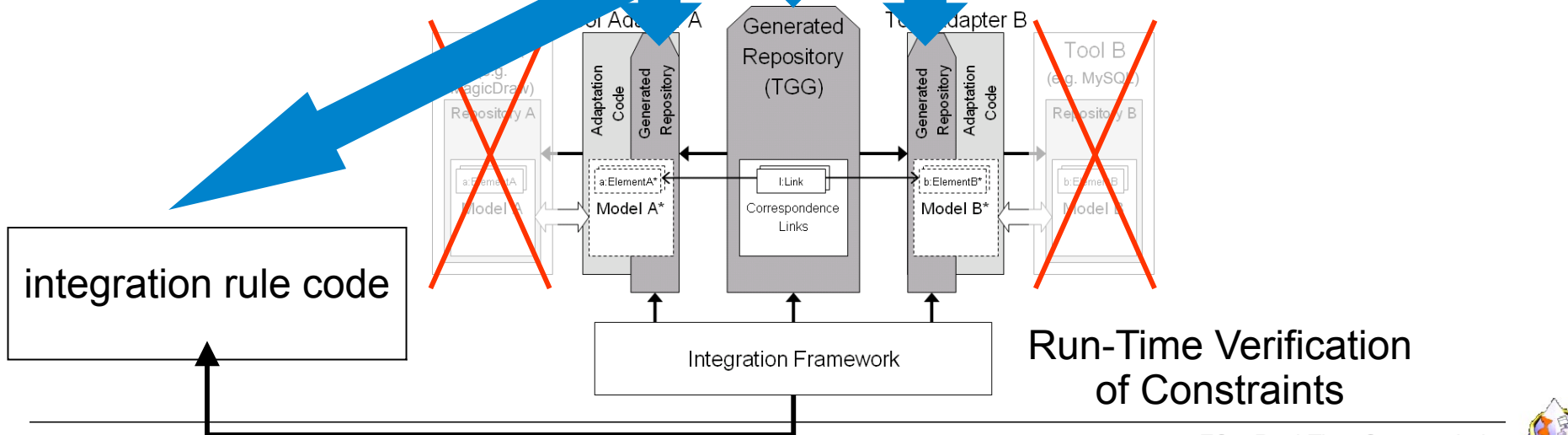
Class Diagrams Metamodel

TGGs

Database Schemata Metamodel

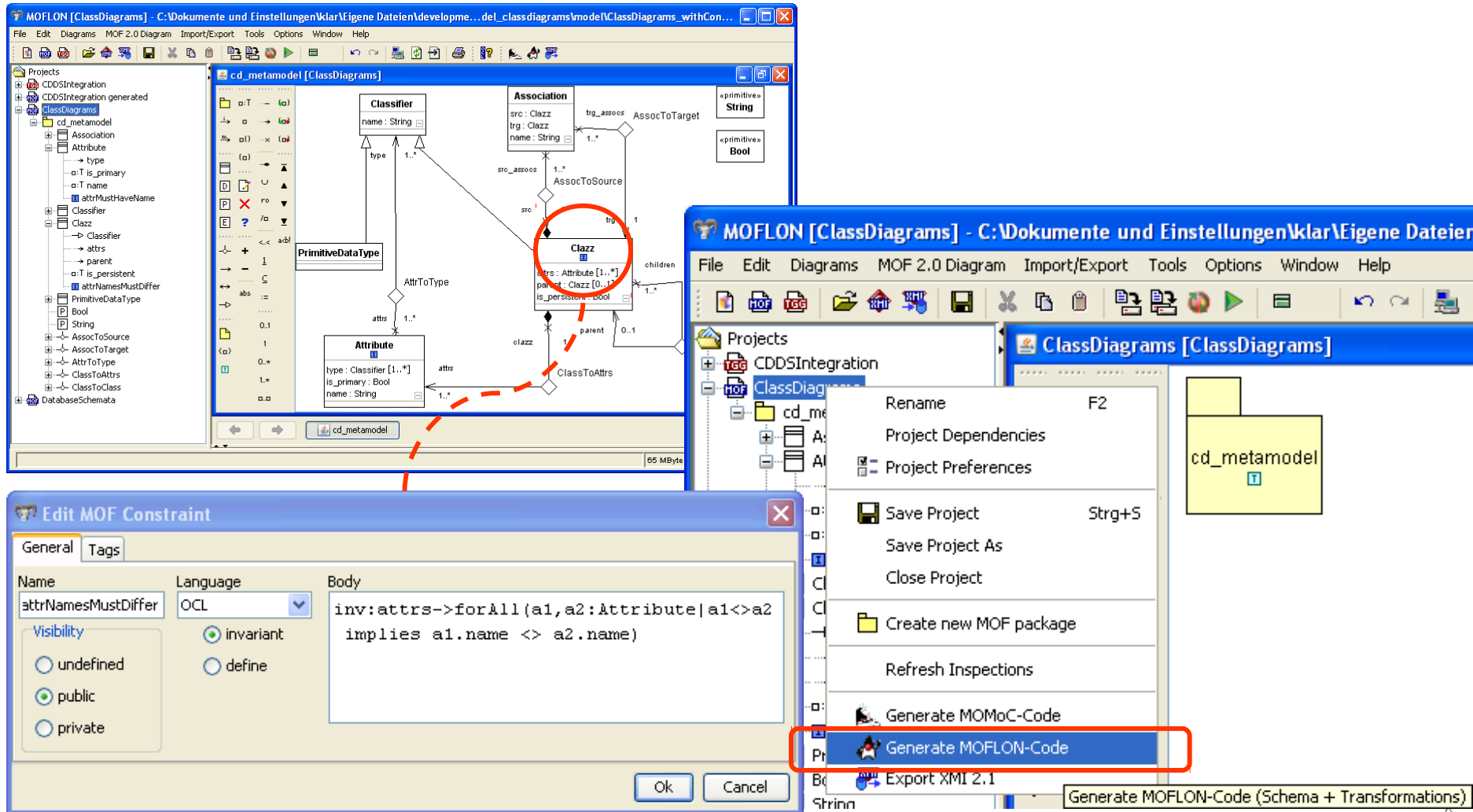


MOFLON  
generates



# TiE-CDDS – Focus on Constraints in CD (1)

## Generate Code from MOF model (CD metamodel)



The screenshot displays the MOFLON [ClassDiagrams] interface. The main window shows a MOF metamodel diagram with classes like Classifier, Association, PrimitiveDataType, and Class. A red circle highlights the 'Class' class. A context menu is open over the 'cd\_metamodel' package, with 'Generate MOFLON-Code' highlighted in red. The 'Edit MOF Constraint' dialog is also visible, showing the constraint 'attrNamesMustDiffer' with the OCL body: `inv:attrs->forAll(a1,a2:Attribute|a1<>a2 implies a1.name <> a2.name)`.

**MOFLON [ClassDiagrams] - C:\Dokumente und Einstellungen\klar\Eigene Dateien\developme...\_del\_classdiagrams\mof\ClassDiagrams\_withCon...**

**MOFLON [ClassDiagrams] - C:\Dokumente und Einstellungen\klar\Eigene Dateien...**

**Edit MOF Constraint**

General | Tags

Name: attrNamesMustDiffer | Language: OCL | Body: inv:attrs->forAll(a1,a2:Attribute|a1<>a2 implies a1.name <> a2.name)

Visibility:  undefined,  public,  private |  invariant,  define

Buttons: Ok, Cancel

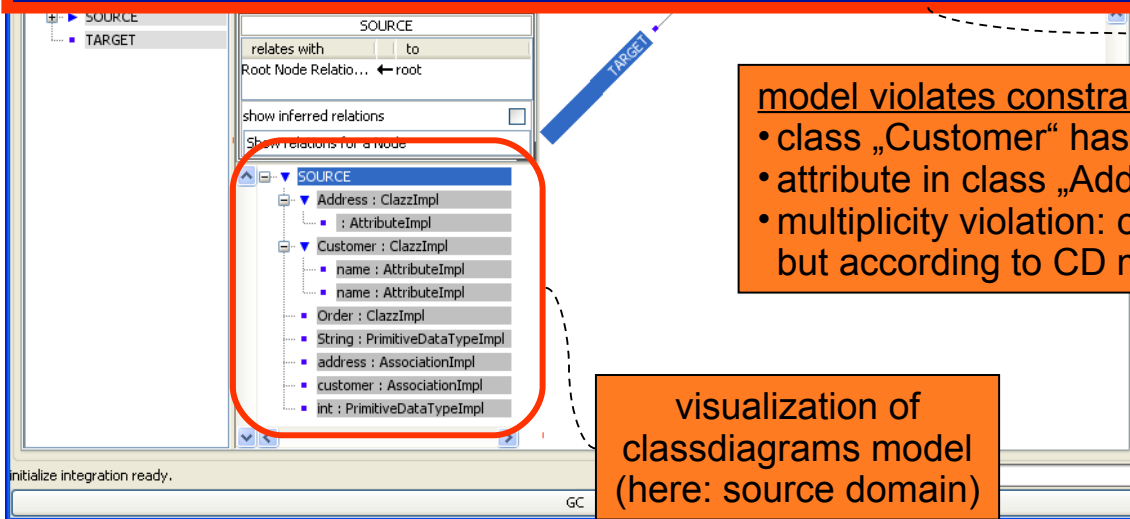
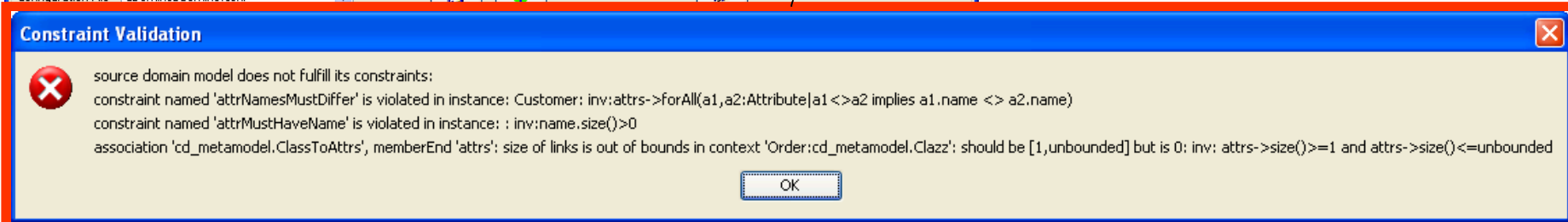
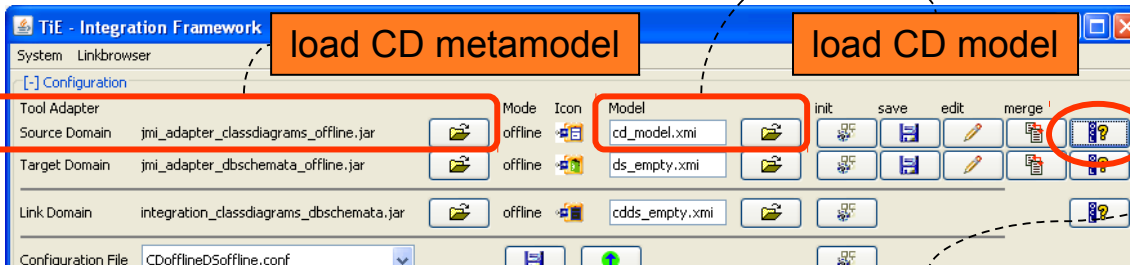
Context Menu: Rename (F2), Project Dependencies, Project Preferences, Save Project (Strg+S), Save Project As, Close Project, Create new MOF package, Refresh Inspections, Generate MOMoC-Code, **Generate MOFLON-Code**, Export XMI 2.1

Generate MOFLON-Code (Schema + Transformations)



# TiE-CDDS – Focus on Constraints in CD (2)

## Integration Framework



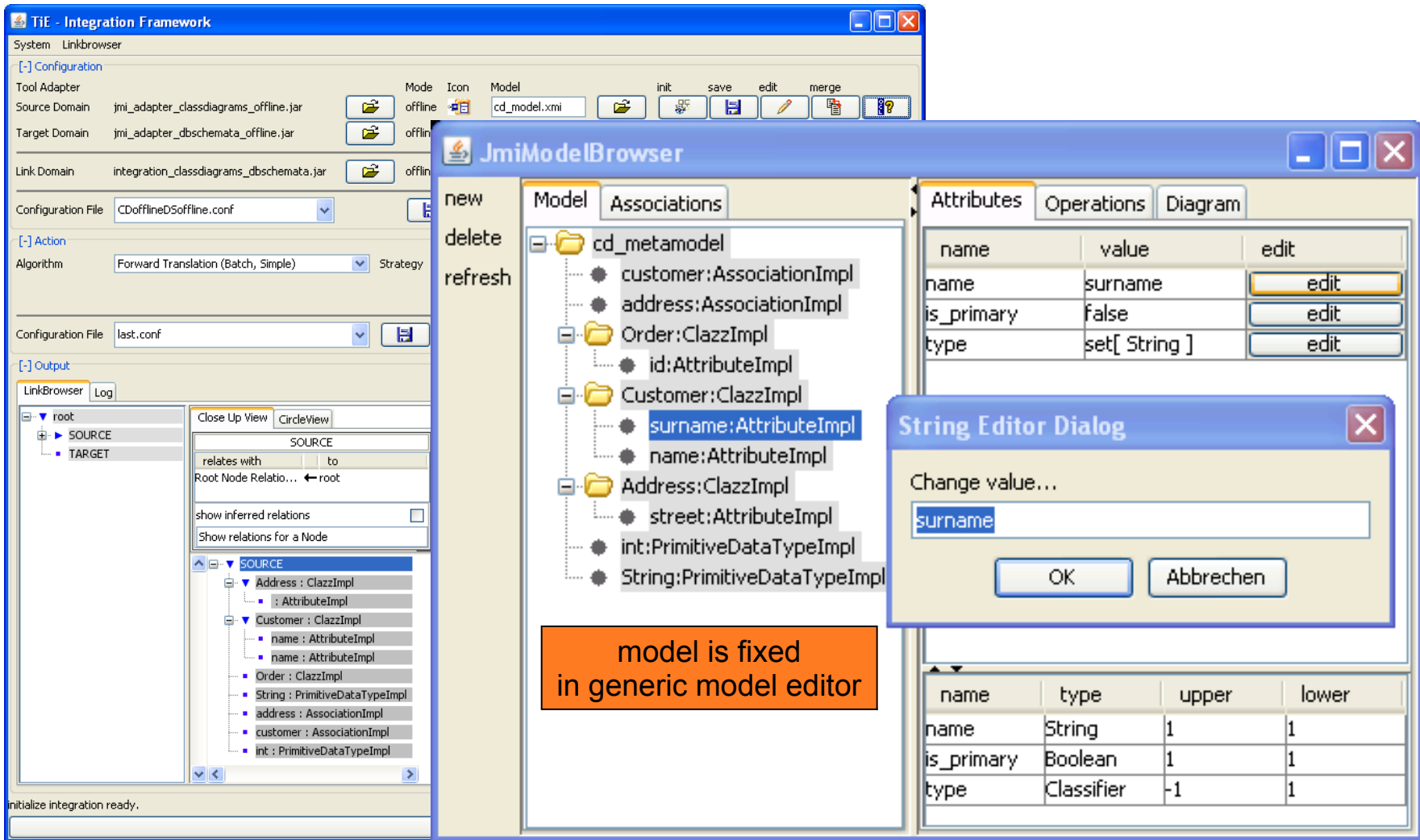
model violates constraints:

- class „Customer“ has two attributes with same name: „name“
- attribute in class „Address“ has no name
- multiplicity violation: class „Order“ has no attribute but according to CD metamodel every class must have one

visualization of  
classdiagrams model  
(here: source domain)



# TiE-CDDS – Focus on Constraints in CD (3) Model Browser



**String Editor Dialog**

Change value...

surname

OK Abbrechen

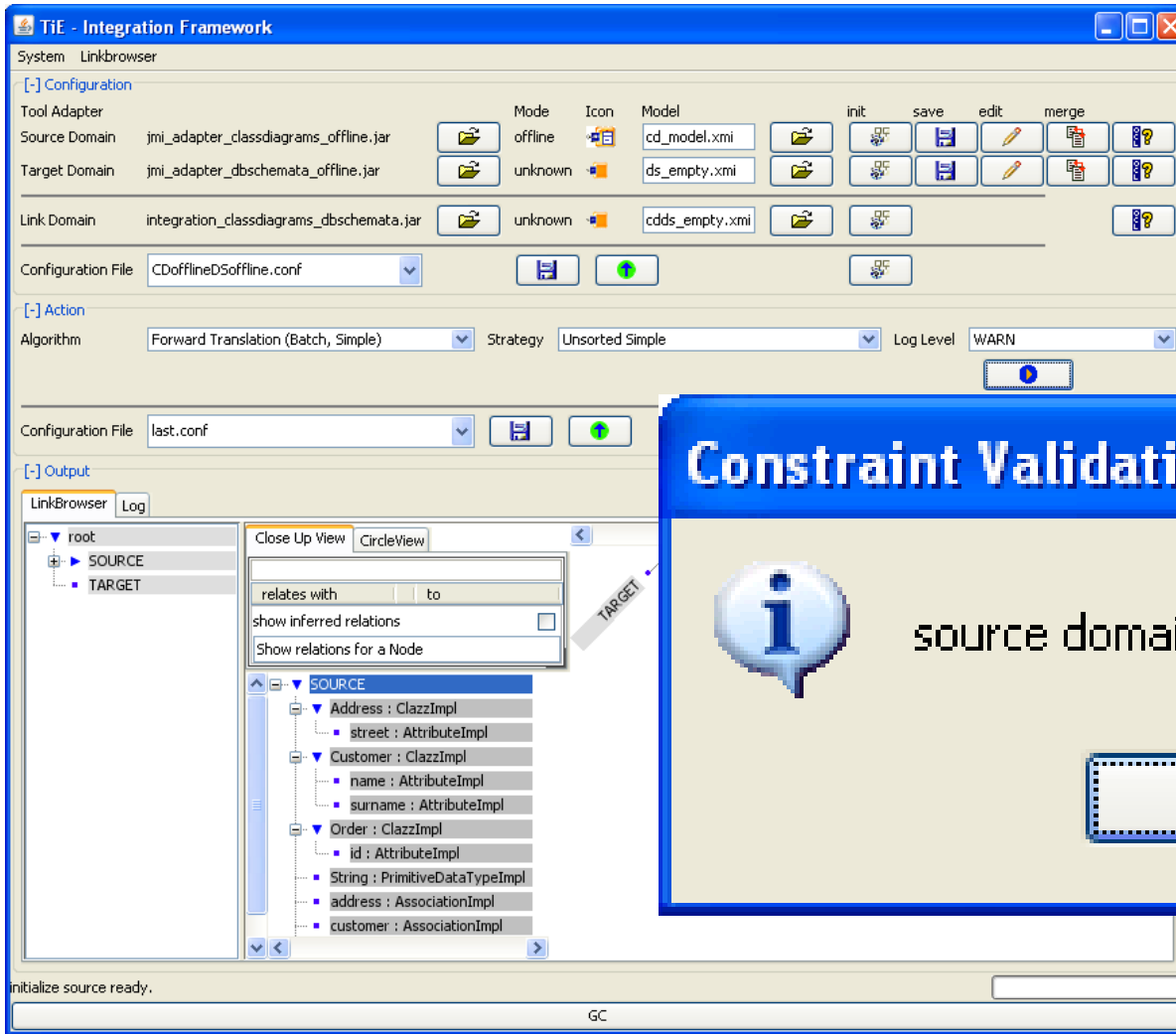
name	value	edit
name	surname	edit
is_primary	false	edit
type	set[ String ]	edit

name	type	upper	lower
name	String	1	1
is_primary	Boolean	1	1
type	Classifier	-1	1

model is fixed  
in generic model editor



# TiE-CDDS – Focus on Constraints in CD (4) Integration Framework



System Linkbrowser

[ - ] Configuration

Tool Adapter	Mode	Icon	Model	init	save	edit	merge
Source Domain jmi_adapter_classdiagrams_offline.jar	offline		cd_model.xmi				
Target Domain jmi_adapter_db schemata_offline.jar	unknown		ds_empty.xmi				
Link Domain integration_classdiagrams_db schemata.jar	unknown		cdds_empty.xmi				

Configuration File CDofflineD5offline.conf

[ - ] Action

Algorithm Forward Translation (Batch, Simple) Strategy Unsorted Simple Log Level WARN

Configuration File last.conf

[ - ] Output

LinkBrowser Log

root

- SOURCE
- TARGET

Close Up View CircleView

relates with to

show inferred relations

Show relations for a Node

SOURCE

- Address : ClassImpl
  - street : AttributeImpl
- Customer : ClassImpl
  - name : AttributeImpl
  - surname : AttributeImpl
- Order : ClassImpl
  - id : AttributeImpl
- String : PrimitiveDataTypeImpl
- address : AssociationImpl
- customer : AssociationImpl

initialize source ready.

GC

translation process  
may start now...

## Constraint Validation

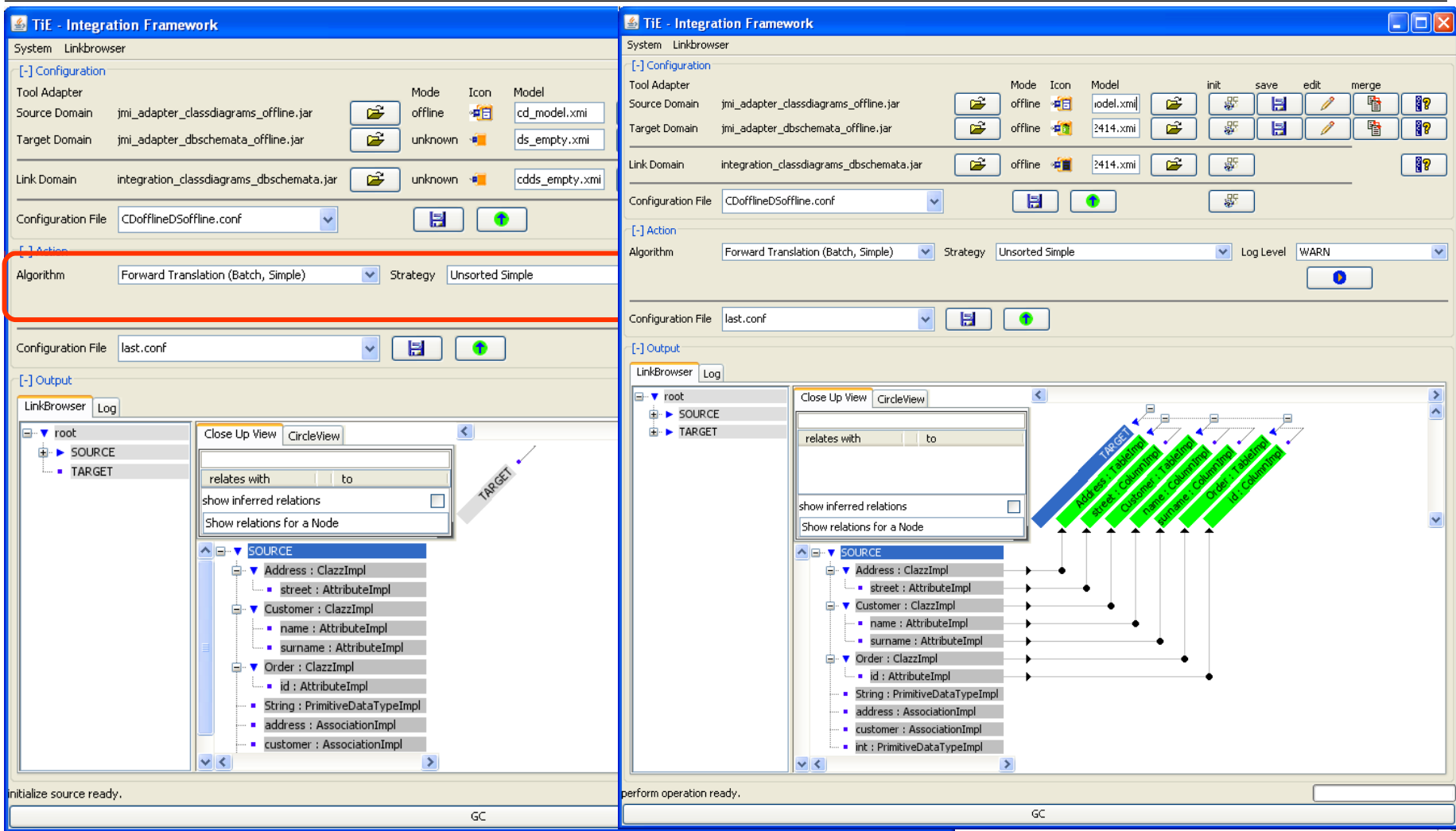


source domain model fulfills its constraints

OK



# TiE-CDDS – Focus on Constraints in CD (5) Forward Translation to DB representation



**TiE - Integration Framework**

System Linkbrowser

[-] Configuration

Tool Adapter	Mode	Icon	Model
Source Domain	jmi_adapter_classdiagrams_offline.jar	offline	cd_model.xmi
Target Domain	jmi_adapter_db schemata_offline.jar	unknown	ds_empty.xmi
Link Domain	integration_classdiagrams_db schemata.jar	unknown	cdds_empty.xmi

Configuration File: CDofflineD5offline.conf

[-] Action

Algorithm: Forward Translation (Batch, Simple) Strategy: Unsorted Simple

Configuration File: last.conf

[-] Output

LinkBrowser Log

Close Up View CircleView

relates with to

show inferred relations

Show relations for a Node

SOURCE

- Address : ClassImpl
  - street : AttributeImpl
- Customer : ClassImpl
  - name : AttributeImpl
  - surname : AttributeImpl
- Order : ClassImpl
  - id : AttributeImpl
- String : PrimitiveDataTypeImpl
- address : AssociationImpl
- customer : AssociationImpl

perform operation ready.

GC



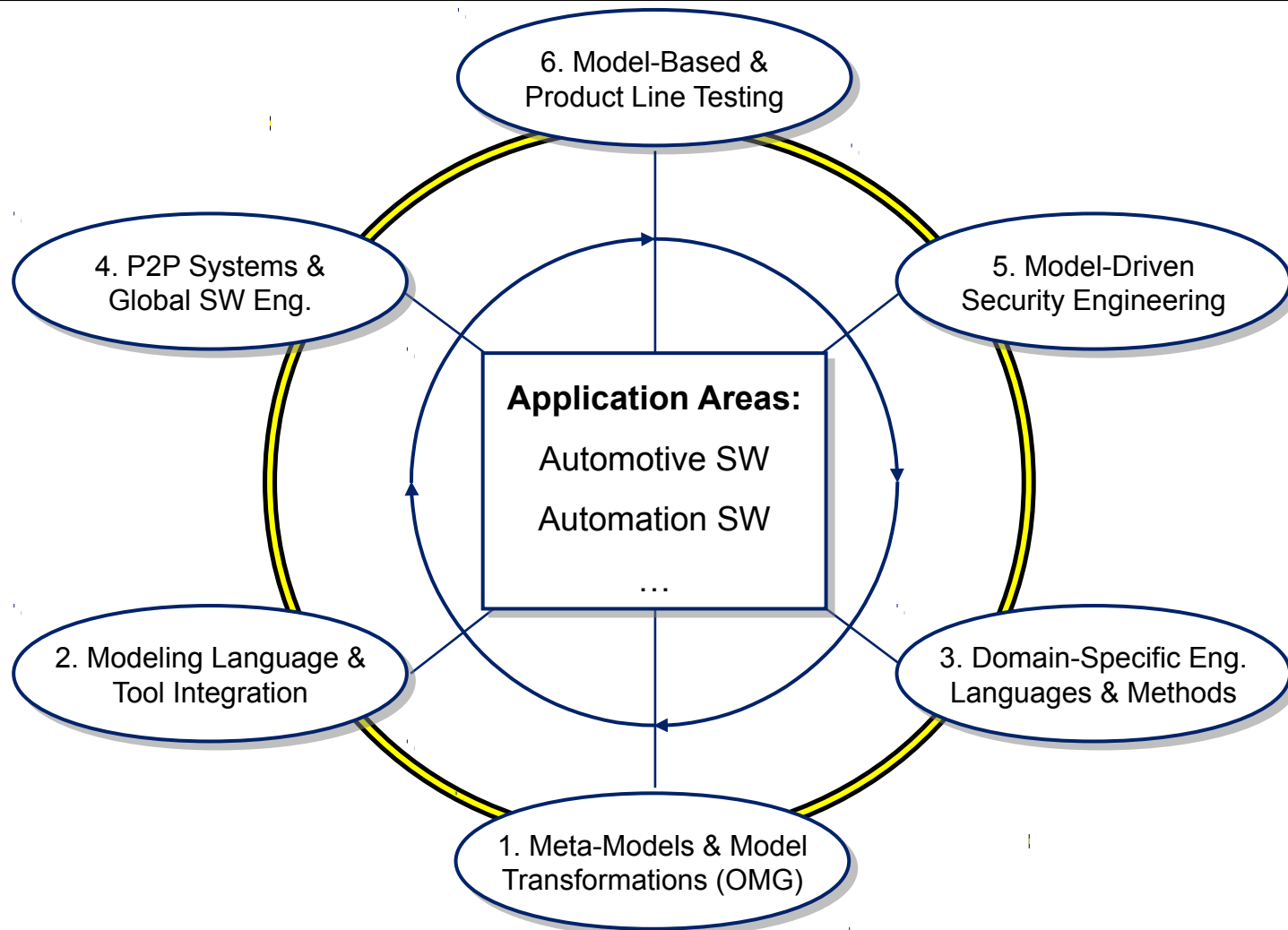


## Future Work – OCL

- Activate more features of Dresden OCL in MOFLON
  - MOF editor
    - User friendly OCL syntax checking
    - OCL expression completion
  - MOFLON code generator
    - Initial Values (init)
  - Queries?
  - ...
  
- We bootstrap our MOFLON MOF Metamodel periodically
  - Add more OCL constraints to our MOF Metamodel
  - Regenerate MOFLON MOF implementation
  - Activate constraint checking in MOFLON
    - Model Verification



# Model-Driven Software Development at Real-Time Systems Lab (Prof. Schürr)



# Related Approaches

standards	approaches based on graph-/modeltransformation					classic meta-CASE approaches				text based approaches					
	MOF, OCL, QVT	Fujaba & MOFLON	Progres & TGG	GME & TGG	EMF & GREAT	EMF & Tefkat	AToM <sup>3</sup>	Microsoft DSL MetaEdit+	EMF & DSL	EMF & GMF	Pounamu	DiaGen	EBNF & TXL	SQL	XML
Abstract syntax	+	+	+	+	o	o	o	+	+	o	+	+	+	o	+
Concrete syntax	--	--	--	+	+	--	+	+	+	+	+	+	+	--	--
Static semantics	+	+	o	+	+	+	o	o	--	+	o	+	o	o	--
Dynamic semantics	+	+	+	+	+	+	+	o	o	--	--	--	+	--	o
Model analysis	+	+	+	+	+	o	+	o	--	+	--	o	+	o	+
Model transformation	+	+	+	+	+	+	+	o	--	--	--	o	+	o	+
Model integration	+	+	+	+	o	+	--	--	--	--	--	--	o	--	o
Acceptability	+	+	o	--	o	+	--	+	--	o	+	o	o	+	+
Scaleability	+	+	--	o	--	o	--	o	--	--	--	--	--	--	o
Tool availability	--	o	o	+	+	+	+	+	o	o	+	+	+	+	o
Expressiveness	+	+	o	+	+	o	o	o	o	o	o	o	o	+	o

from Amelunxen, Königs, Rötschke, and Schürr,  
**„MOSL: Composing a Visual Language for a Metamodeling Framework“**  
 in IEEE Symposium on Visual Languages and Human-Centric Computing (VLHCC 2006),  
 September, 2006, 81-84

## Further reading

- **A. Königs, A. Schürr: "Tool Integration with Triple Graph Grammars - A Survey", in: R. Heckel (ed.), Proceedings of the SegraVis School on Foundations of Visual Modelling Techniques, Amsterdam: Elsevier Science Publ., 2006; Electronic Notes in Theoretical Computer Science, Vol. 148, 113-150.**
- **F. Klar, S. Rose, A. Schürr: "TiE - A Tool Integration Environment", Proceedings of the 5th ECMDA Traceability Workshop, 2009; CTIT Workshop Proceedings, Vol. WP09-09, 39-48**
- **F. Klar, S. Rose, A. Schürr: "A Meta-Model-Driven Tool Integration Development Process", Proceedings of the 2nd International United Information Systems Conference, 2008; Lecture Notes in Business Information Processing, 201-212.**
- **C. Amelunxen, A. Königs, T. Röttschke, A. Schürr: "MOFLON: A Standard-Compliant Metamodeling Framework with Graph Transformations", in: A. Rensink, J. Warmer (eds.), Model Driven Architecture - Foundations and Applications: Second European Conference, Heidelberg: Springer Verlag, 2006; Lecture Notes in Computer Science (LNCS), Vol. 4066, Springer Verlag, 361-375.**
- **A. Königs: "Model Integration and Transformation - A Triple Graph Grammar-based QVT Implementation", Technische Universität Darmstadt, Phd Thesis, 2009.**





# Time for questions and discussion

Thank you for your attention...



<http://www.moflon.org>

