

45. How to Synchronize Models with Triple Graph Grammars for Data Connection

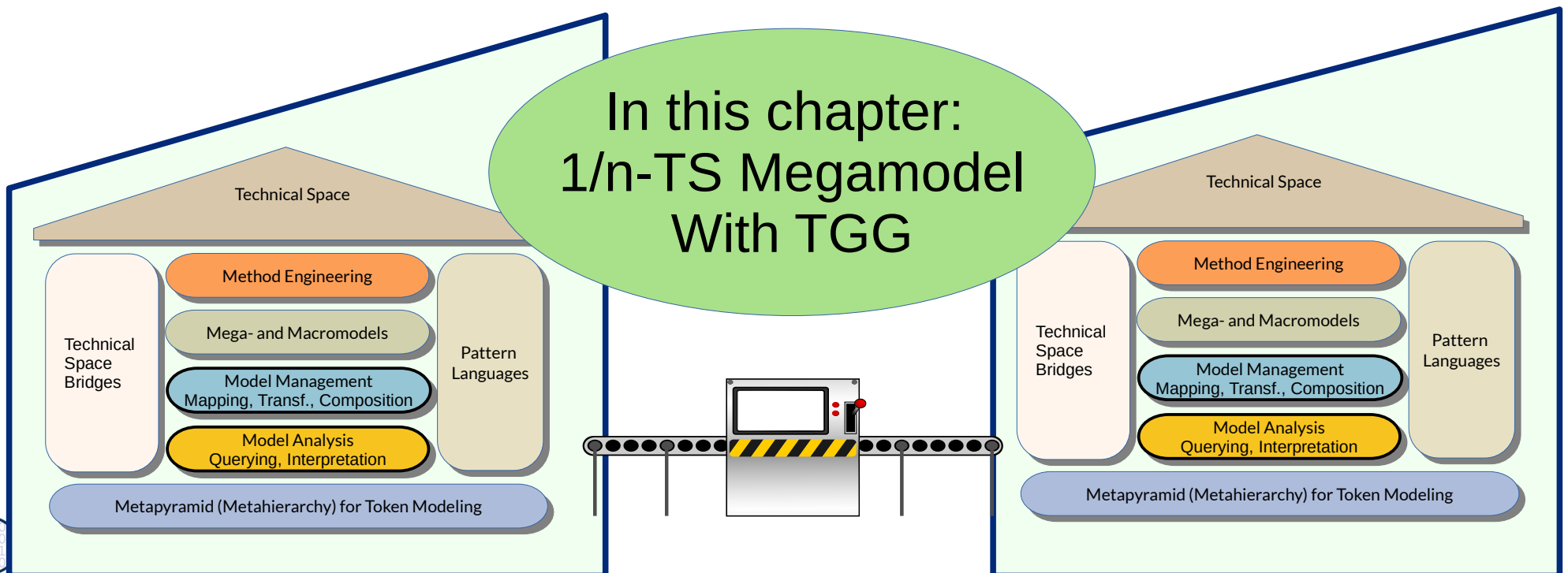
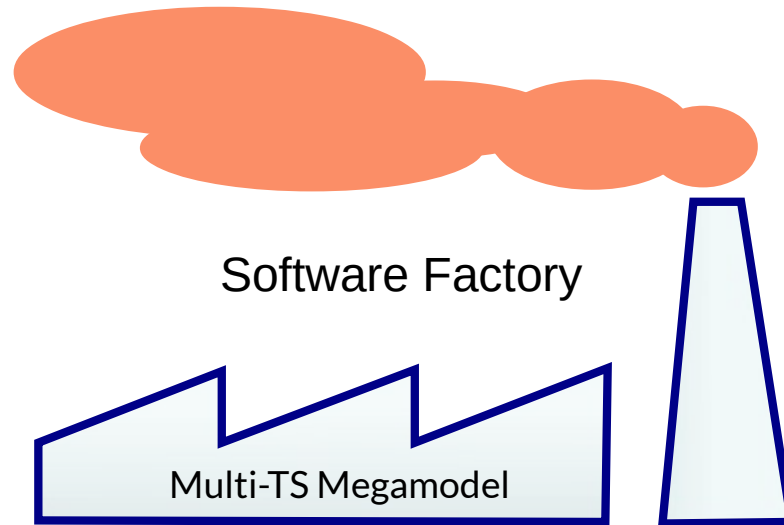
Prof. Dr. U. Aßmann
Technische Universität Dresden
Institut für Software- und
Multimediatechnik
Gruppe Softwaretechnologie
<http://st.inf.tu-dresden.de>
Version 19-0.6, 27.01.20

- 1) Triple Graph Grammars
- 2) TGG in MOFLON
- 3) Using TGG in MOFLON



- ▶ Anthony Anjorin, Erhan Leblebici, and Andy Schürr. 20 years of triple graph grammars: A roadmap for future research. ECEASST, 73, 2015.
- ▶ F. Klar, A. Königs, A. Schürr: "Model Transformation in the Large", Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on the foundations of software engineering, New York: ACM Press, 2007; ACM Digital Library Proceedings, 285-294. <http://www.idt.mdh.se/esec-fse-2007/>
- ▶ www.fujaba.de
- ▶ www.moflon.org, <https://emoflon.org/>
 - <https://paper.dropbox.com/doc/Meta-Modelling-with-eMoflonCore--ArVO3r~~geAdwkL9vVBUTzKZAg-zyOqELGZ0X9jL85TAs7pf>
- ▶ T. Fischer, J. Niere, L. Torunski, and A. Zündorf, 'Story Diagrams: A new Graph Rewrite Language based on the Unified Modeling Language', in Proc. of the 6th International Workshop on Theory and Application of Graph Transformation (TAGT), Paderborn, Germany (G. Engels and G. Rozenberg, eds.), LNCS 1764, pp. 296--309, Springer Verlag, November 1998.
<http://www.upb.de/cs/ag-schaefer/Veroeffentlichungen/Quellen/Papers/1998/TAGT1998.pdf>

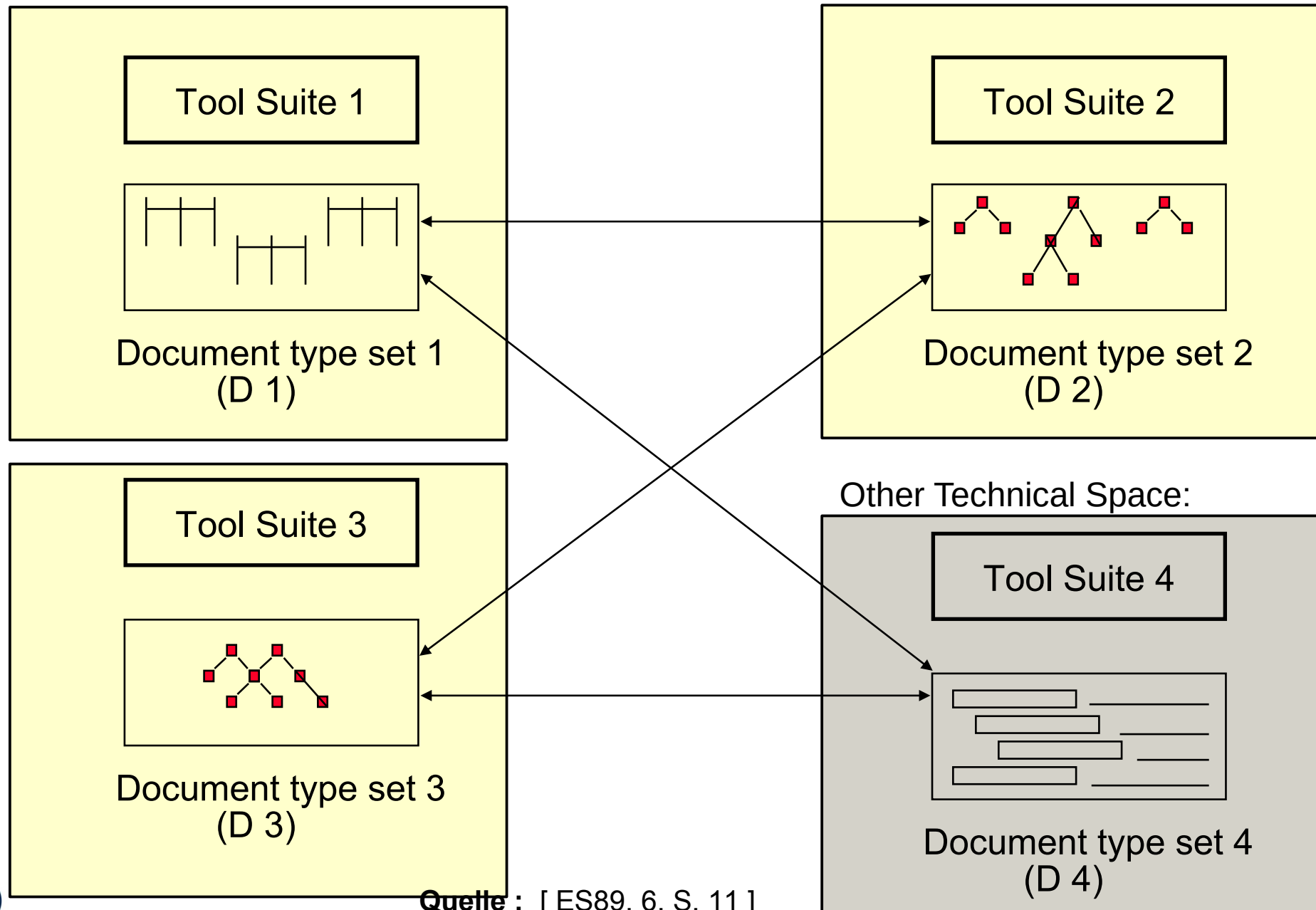
Q11: A Software Factory's Heart: the Multi-TS Megamodel



Integration of Tool Suites by Data Connection

5

Model-Driven Software Development in Technical Spaces (MOST)



Quelle : [ES89, 6, S. 11]

45.1 „Synchronizing“ models with Triple Graph Grammars

Mapping graphs to other graphs

Specification of mappings with mapping rules

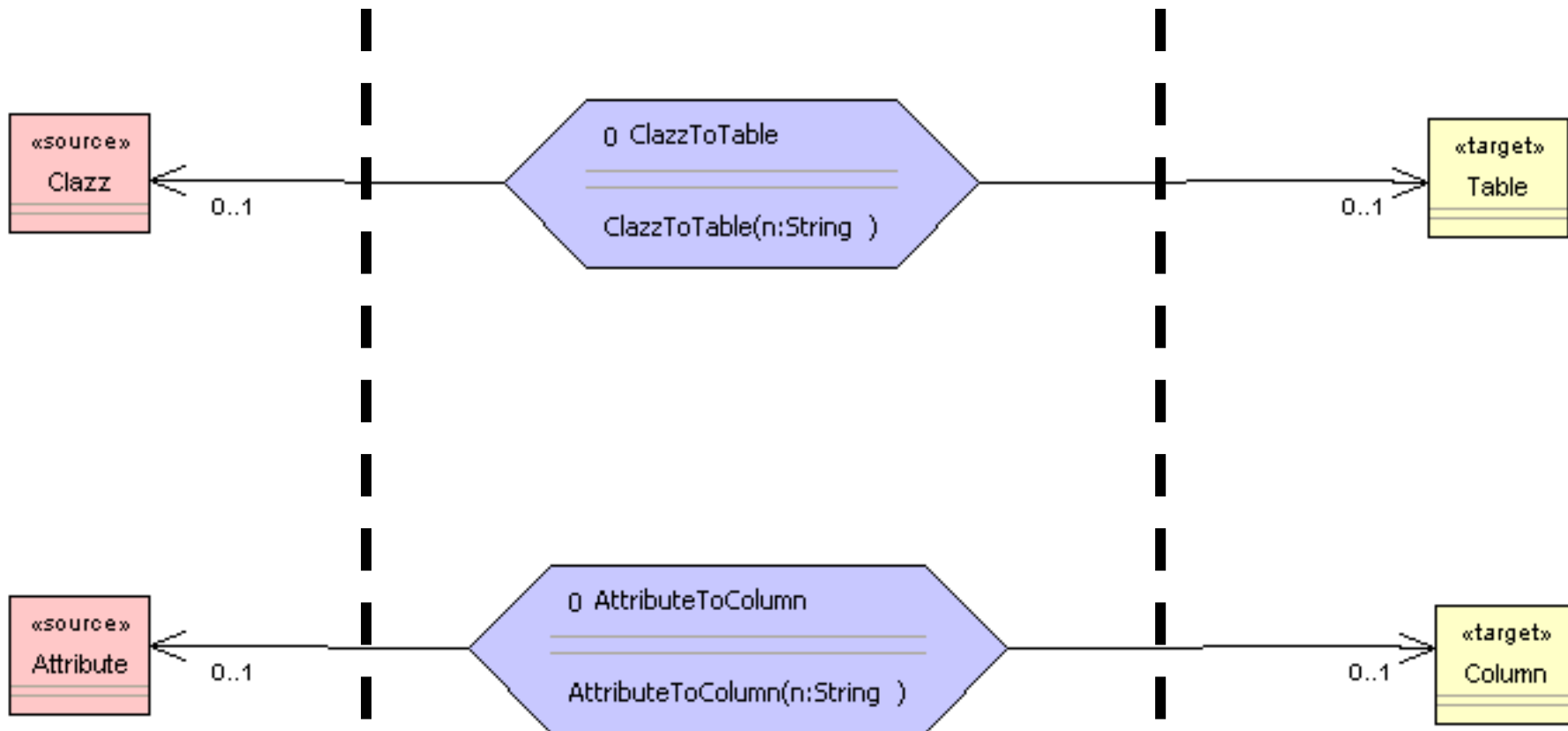
Incremental transformation

Traceability



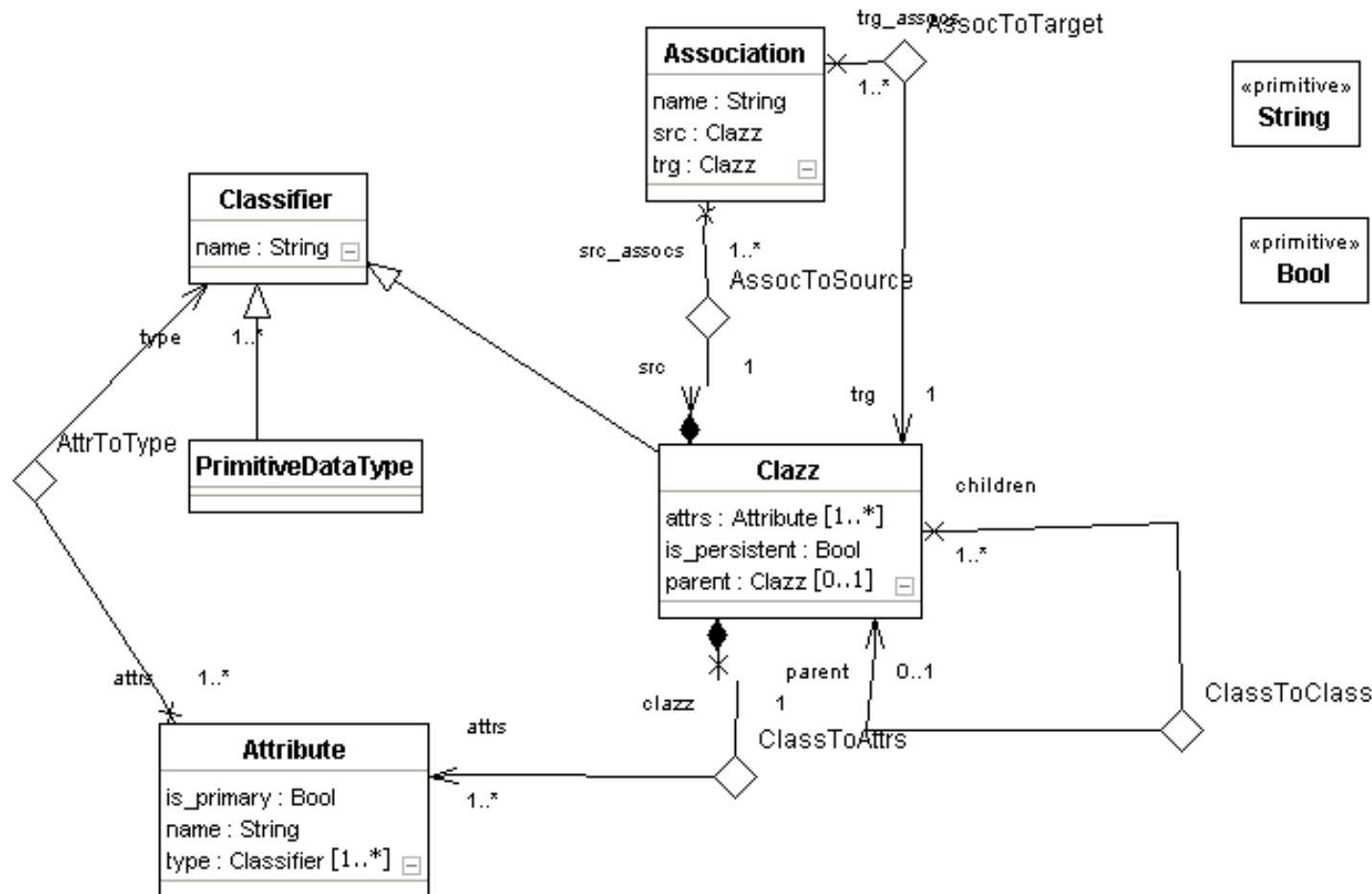
Triple Graph Grammars – Moflon Example

- ▶ A **Triple Graph Grammar (TGG)** is a mapping-oriented transformation system, consisting of rules with three „areas“ (better called **metamodel mapping grammars**)
 - Left side: (source) graph pattern 1 in (source) graph 1
 - Right side: (target) graph pattern 2 in (target) graph 2
 - Middle: relational expression (net) relating graph pattern 1 and 2 (trace model)

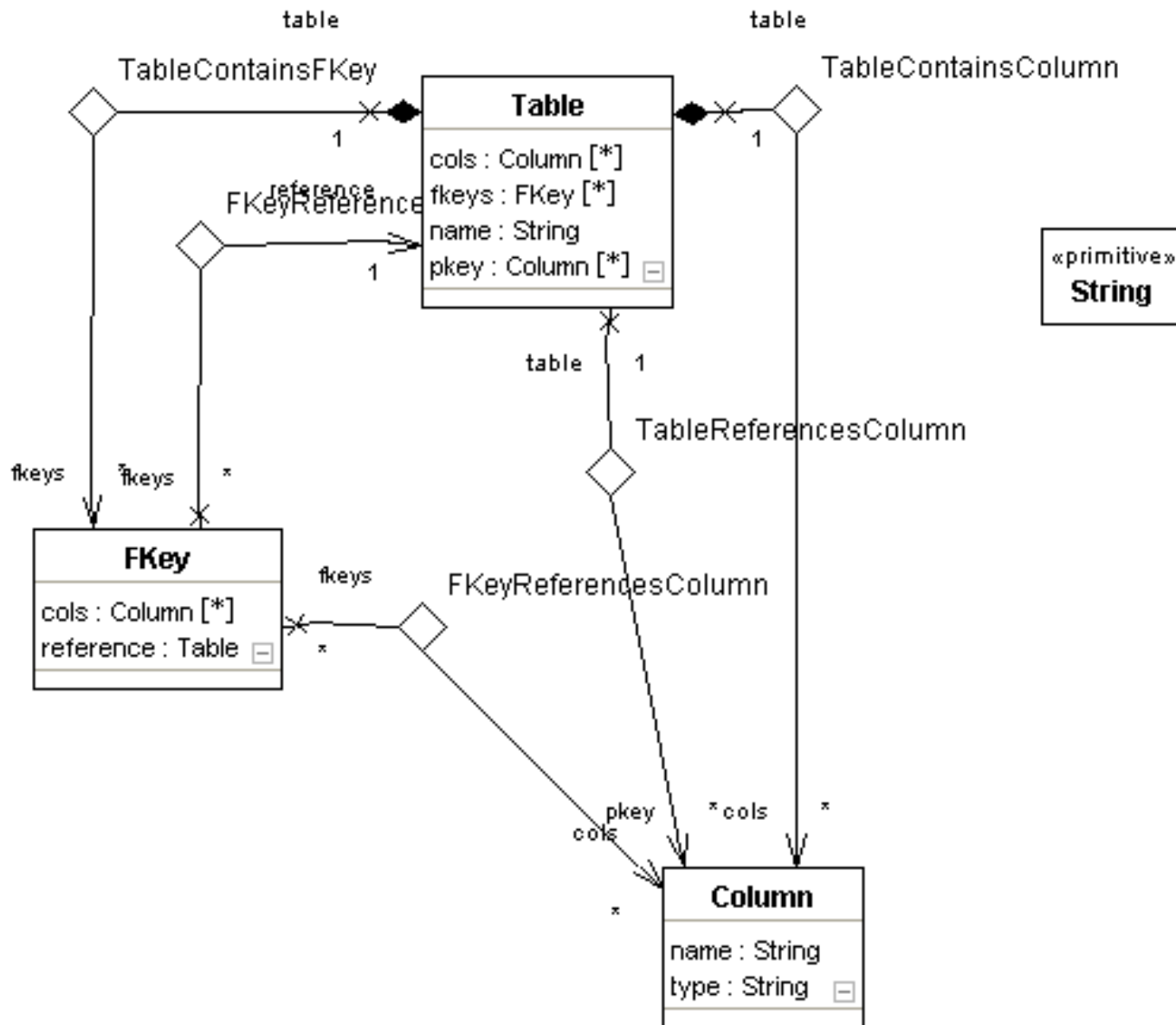


Triple Graph Grammars – Class Diagram Metamodel (CD)

- ▶ Synchronize Class-Diagram-metamodel (CD) with a relational schema (RS): object-relational mapping (ORM)

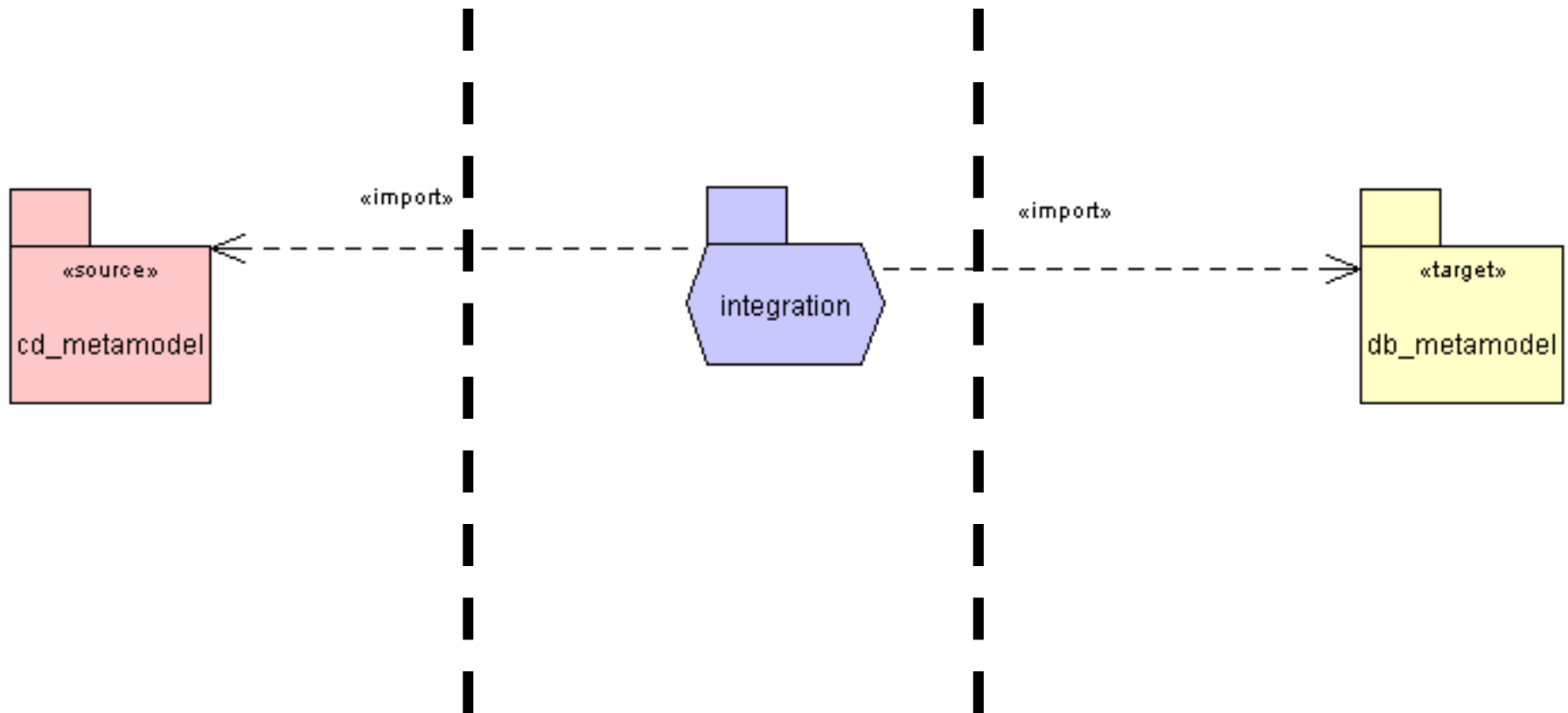


Relational Metamodel (DB, relational schema)



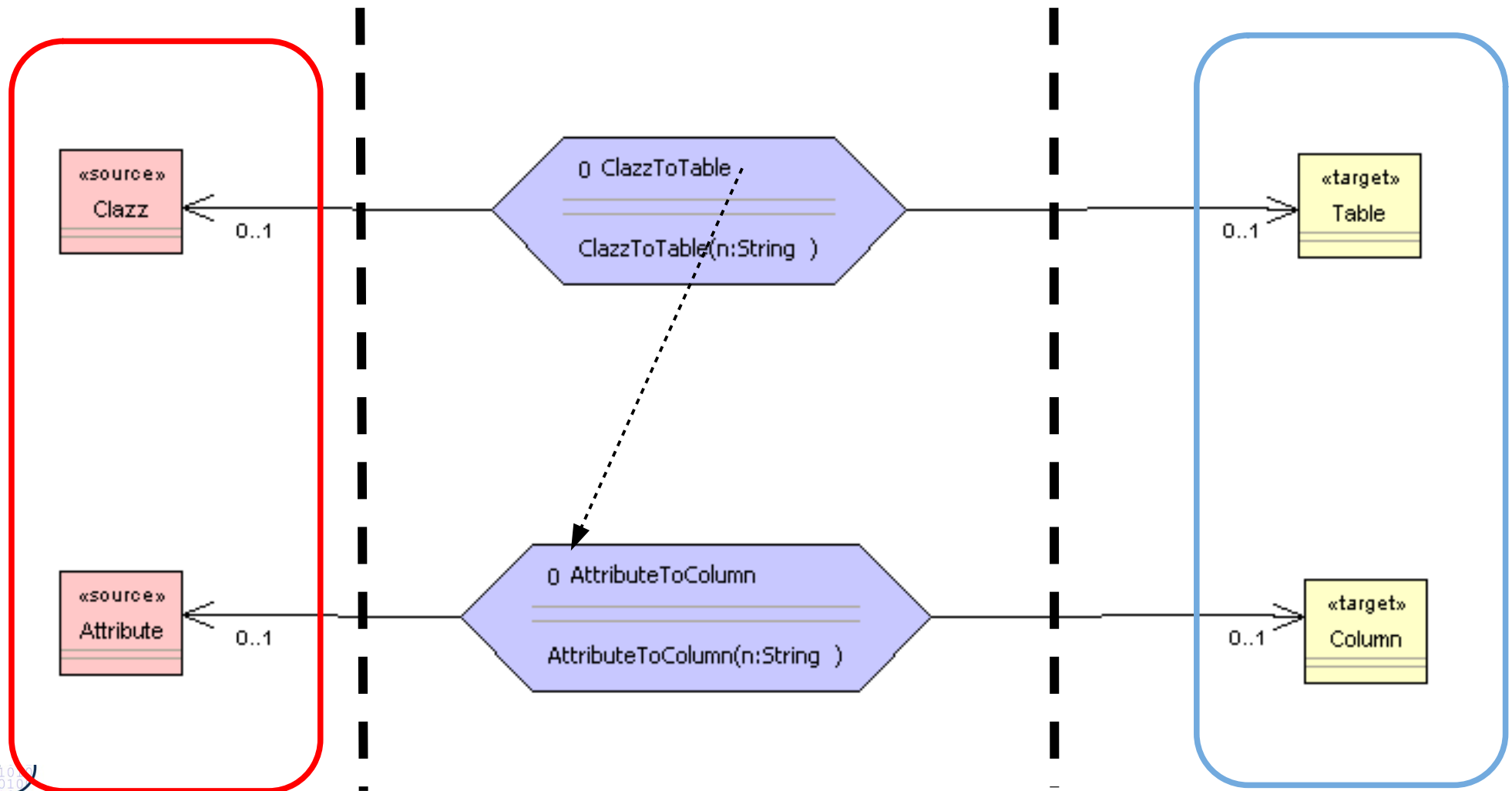
Triple Graph Grammars – Example

- ▶ The metamodel mapping grammar of a TGG has a top rule (start rule) which describes the relationship of the graphs on topmost level



Triple Graph Grammars – Example

- ▶ From the top-rule, other TGG rules are associated („called“)
- ▶ In this case, the TGG only checks (black color – TEST)
- ▶ What happens, if both sides are in different Technical Spaces?



TGG Specify Transformation Bridges Between Roles and Technical Spaces

- ▶ TGG can also be used to synchronize Material roles
 - between two material objects
 - in different repositories
 - even in different technical spaces
- ▶ Only assumption: 1:1 mappings of model elements

TGG are a fine technique to build *transformation bridges for data connection* between tools, even in different technical spaces.

45.2. Triple Graph Grammars in MOFLON

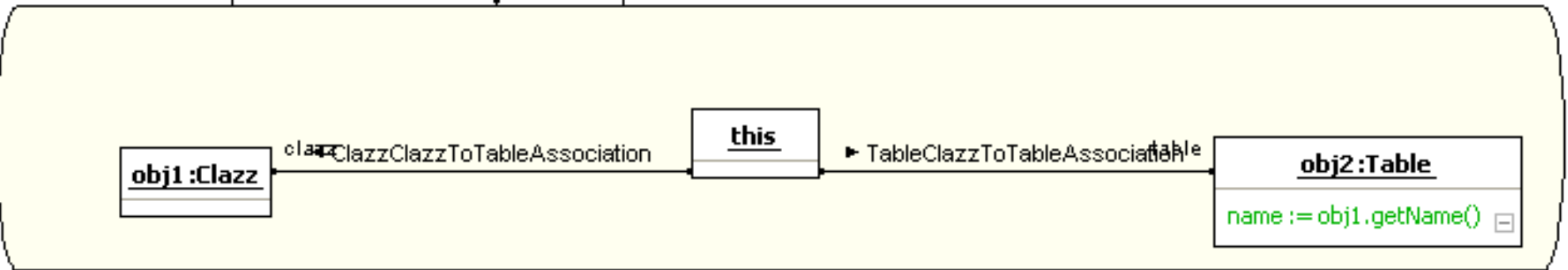
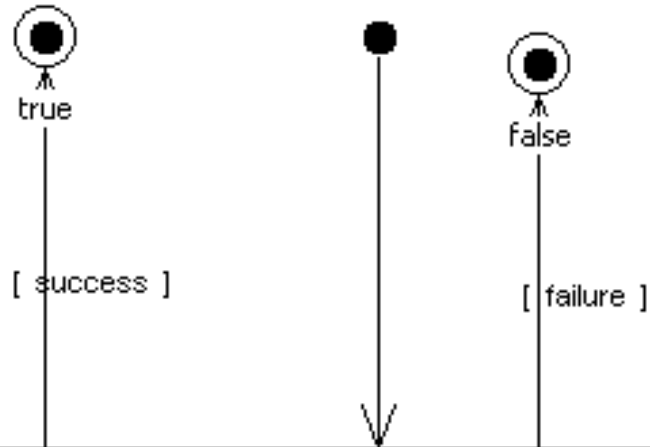
- ▶ MOFLON in MOF Technical Space
- ▶ eMOFLON in EMOF TS



Triple Graph Grammars – Moflon Example

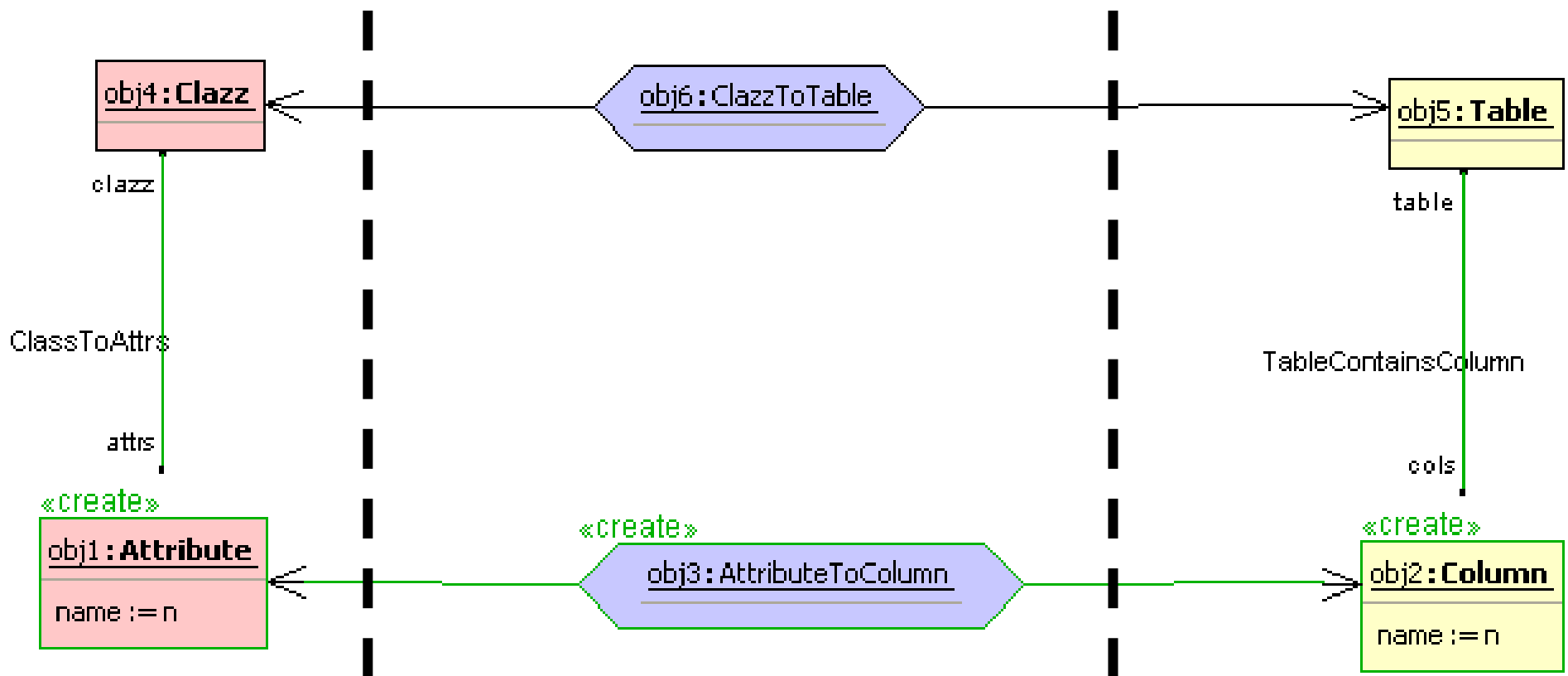
- ▶ Because they are named, TGG rules can be started by Fujaba Storyboards

ClazzToTable::performForwardAttributeValuePropagation(): Boolean

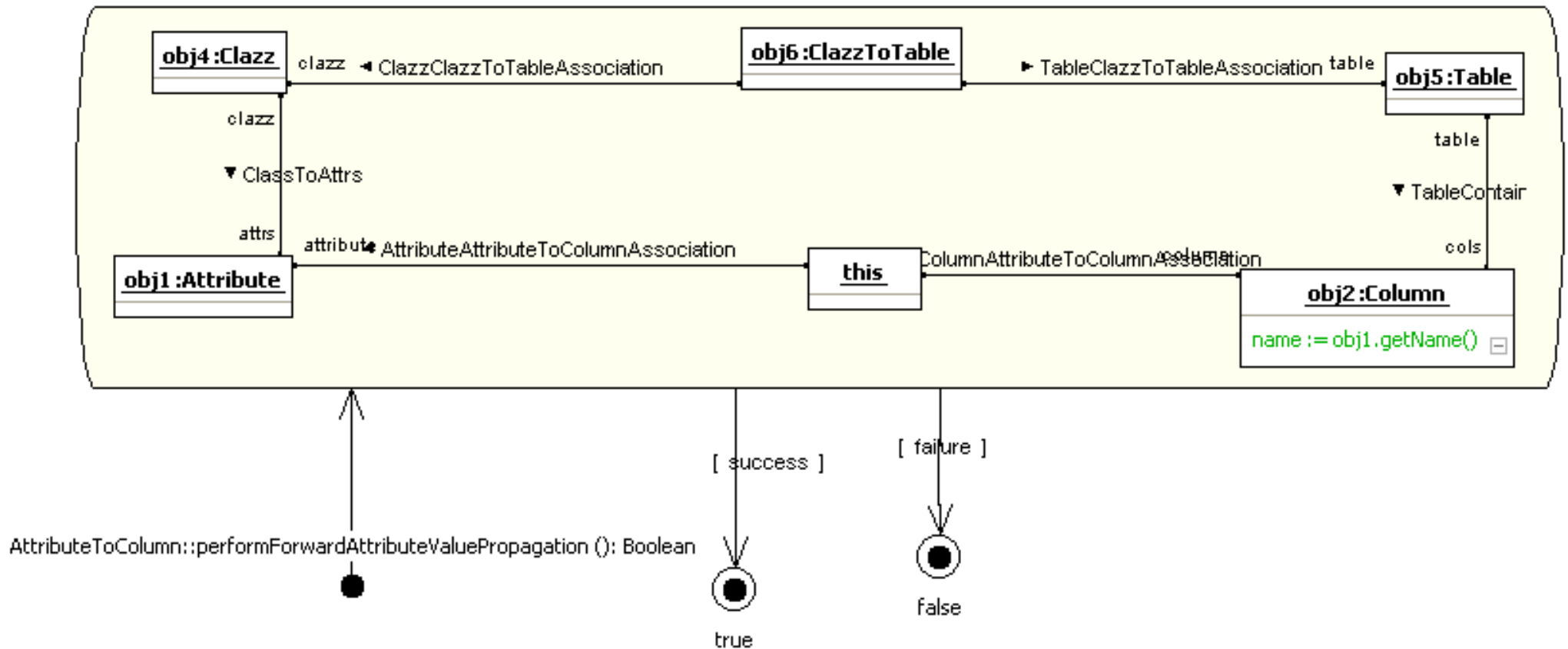


Triple Graph Grammars – Moflon Example

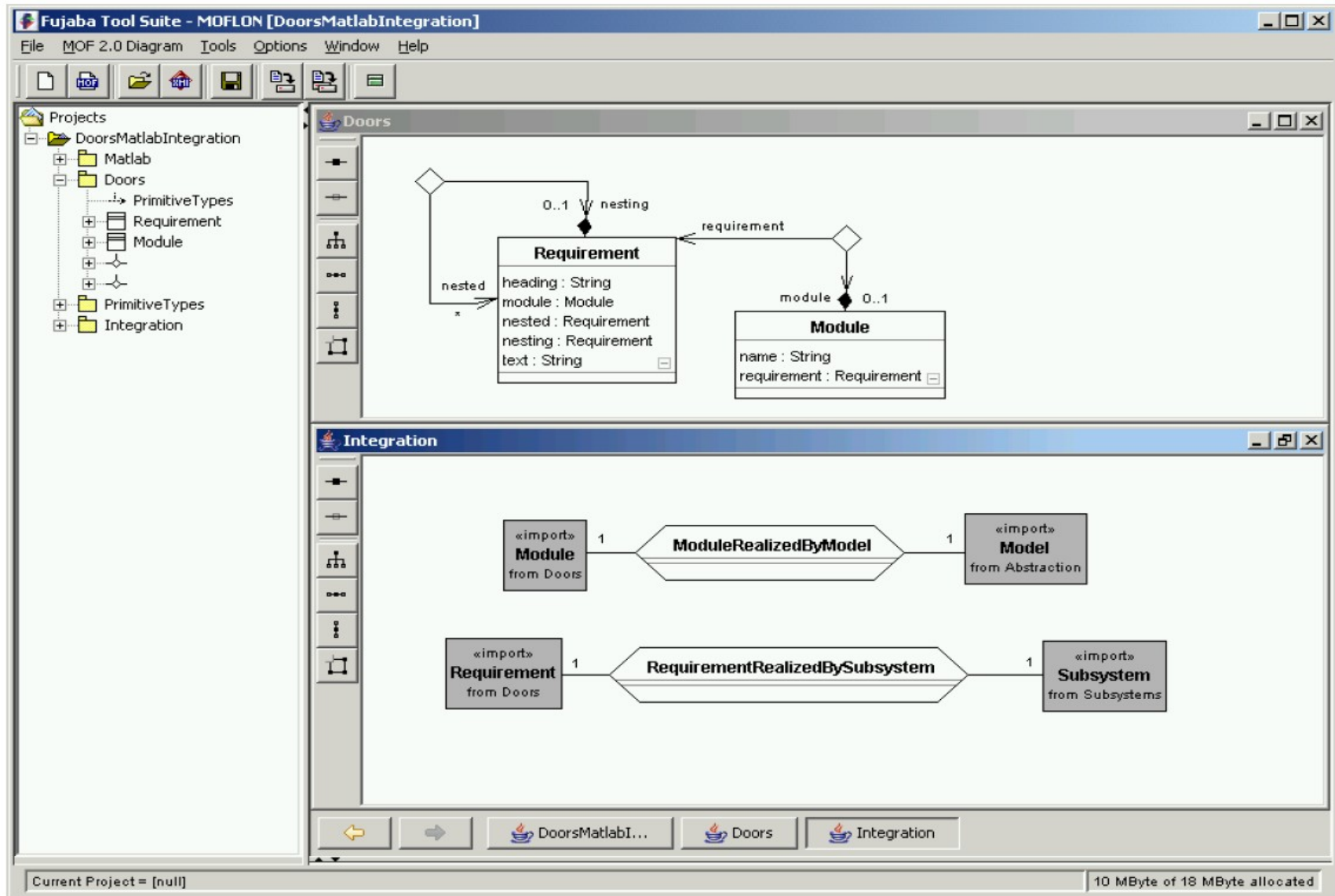
- ▶ Pairwise correspondance of model elements on both sides
- ▶ Here, objects on the lower level are created anew



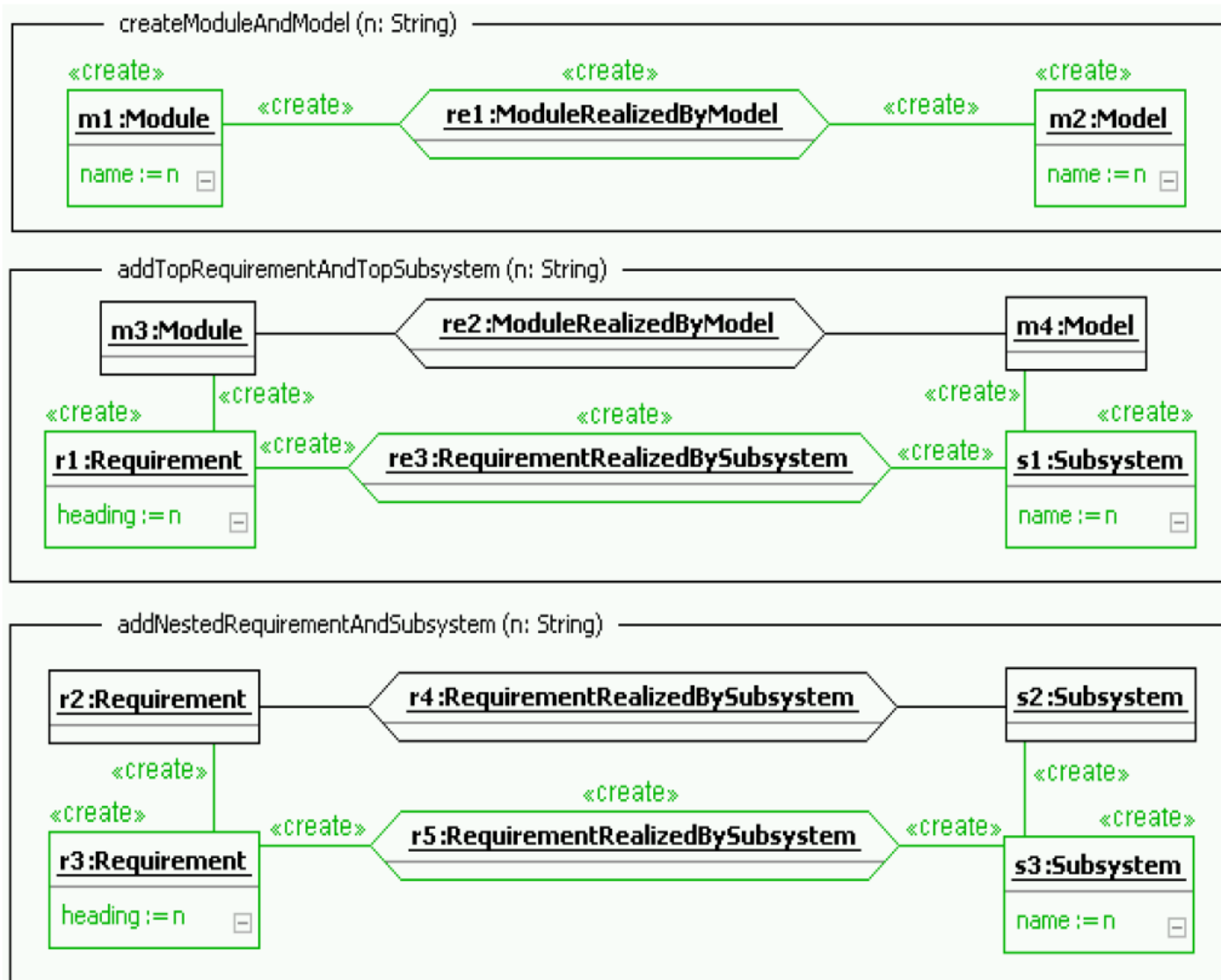
Triple Graph Grammars – Moflon Example



Ex. 2: TGG Coupling of Requirements Specification and Design



TGG Coupling Requirements Specification and Design



45.3. Using Triple Graph Grammars in MOFLON

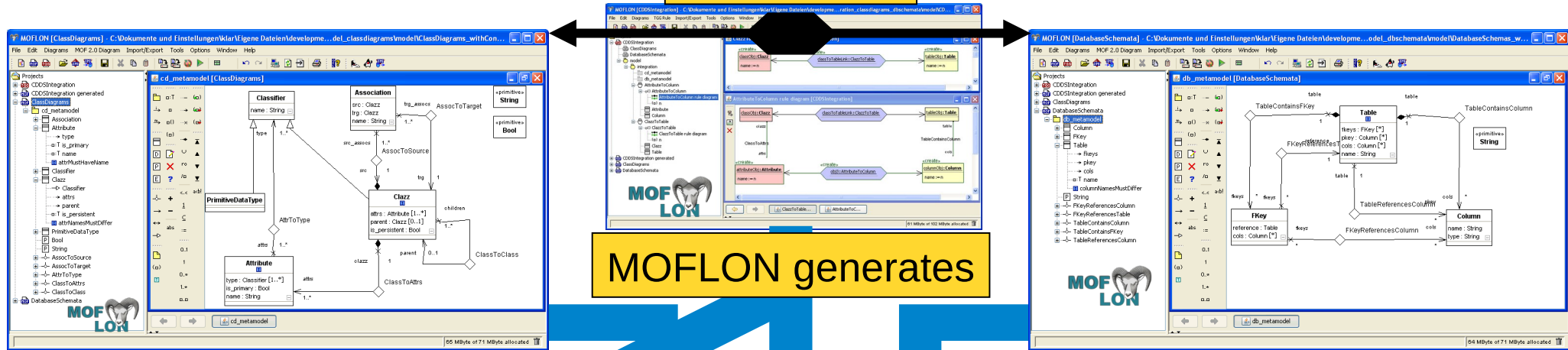


Example: Tool Integration Scenario TiE-CD-DB: (ClassDiagrams / DatabaseSchema)

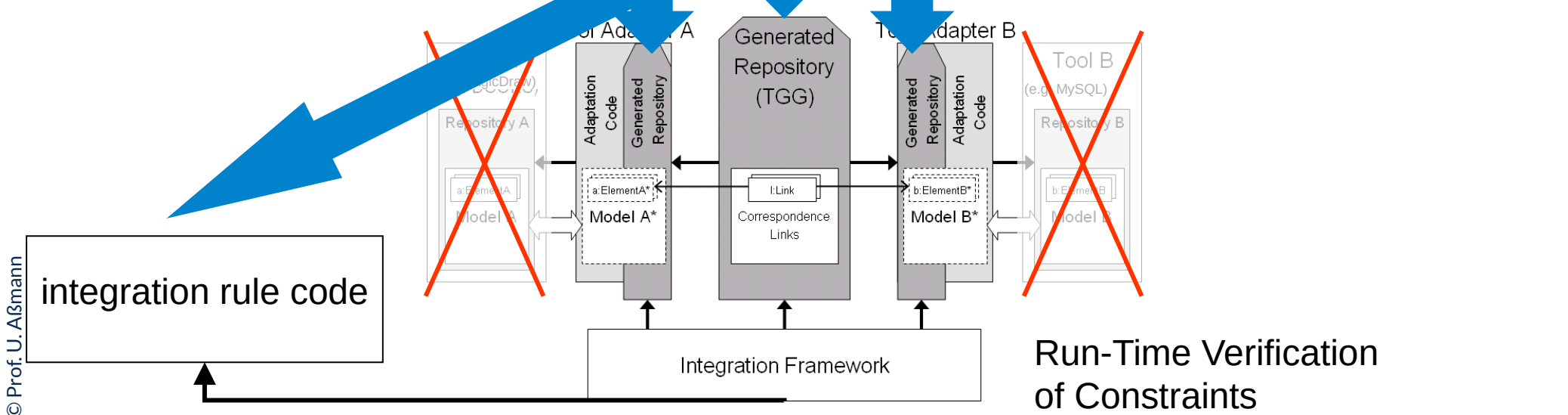
Class Diagrams Metamodel

TGGs relate

Database Schemata Metamodel



MOFLON generates



TiE-CD-DB – Constraints in Class Diagrams (1)

Generate Code from MOF model (CD metamodel)

23

Model-Driven Software Development in Technical Spaces (MOST)

The screenshot displays the MOFLON [ClassDiagrams] environment. The main window shows a class diagram for a CD metamodel with classes: Classifier, Association, PrimitiveDataType, Attribute, andClazz. The classClazz is circled in red. A project browser on the left shows the project structure. A context menu is open over the project browser, with the option 'Generate MOFLON-Code' highlighted in red. An 'Edit MOF Constraint' dialog is also visible, showing the constraint 'attrNamesMustDiffer' with an 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\mode\ClassDiagrams_withCon...

File Edit Diagrams MOF 2.0 Diagram Import/Export Tools Options Window Help

Projects

- CDSDIntegration
- CDSDIntegration generated
- ClassDiagrams
 - cd_metamodel
 - Association
 - Attribute
 - Classifier
 - Clazz
 - PrimitiveDataType

cd_metamodel [ClassDiagrams]

Classifier name: String

Association src:Clazz trg:Clazz name: String

PrimitiveDataType

Attribute type: Classifier [1..*] is_primary: Bool name: String

Clazz attrs: Attribute [1..*] parent:Clazz [0..1] is_persistent: bool

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

File Edit Diagrams MOF 2.0 Diagram Import/Export Tools Options Window Help

Projects

- CDSDIntegration
- ClassDiagrams
 - cd_me
 - As
 - Al

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

ClassDiagrams [ClassDiagrams]

cd_metamodel

Edit MOF Constraint

General Tags

Name: attrNamesMustDiffer

Language: OCL

Body: `inv:attrs->forAll(a1, a2:Attribute | a1 <> a2 implies a1.name <> a2.name)`

Visibility: public

undefined define

Ok Cancel

Generate MOFLON-Code (Schema + Transformations)

TiE-CD-DB – Constraints in Class Diagrams (2)

Integration Framework

Constraint Validation

source domain model does not fulfill its constraints:

- constraint named 'attrNamesMustDiffer' is violated in instance: Customer: inv:attrs->forall(a1,a2:Attribute|a1 <>a2 implies a1.name <> a2.name)
- constraint named 'attrMustHaveName' is violated in instance: : inv:name.size()>0
- association 'cd_metamodel.ClassToAttrs', memberEnd 'attrs': size of links is out of bounds in context 'Order:cd_metamodel.Clazz': should be [1,unbounded] but is 0: inv: attrs->size()>=1 and attrs->size()<=unbounded

OK

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-CD-DB – Constraints in Class Diagrams (3)

Model Browser

25

Model-Driven Software Development in Technical Spaces (MOST)

© Prof. U. Aßmann

model is fixed in generic model editor

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

TiE-CD-DB – Constraints in Class Diagrams (4) Integration Framework

26

Model-Driven Software Development in Technical Spaces (MOST)

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_dbschemata_offline.jar	unknown		ds_empty.xmi				
Link Domain: integration_classdiagrams_dbschemata.jar	unknown		cdds_empty.xmi				

Configuration File: CDOfflineDSOffline.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-CD-DB – Constraints in Class Diagrams (5)

Forward Translation to DB representation

27

Model-Driven Software Development in Technical Spaces (MOST)

The image displays two screenshots of the TiE - Integration Framework software interface, illustrating the forward translation process from a class diagram to a database representation.

Left Screenshot: Configuration and Action Settings

- System Linkbrowser:** Shows configuration for Source Domain (jmi_adapter_classdiagrams_offline.jar), Target Domain (jmi_adapter_dbschemata_offline.jar), and Link Domain (integration_classdiagrams_dbschemata.jar).
- Configuration File:** CDOfflineDSOffline.conf
- Action:** Algorithm: Forward Translation (Batch, Simple); Strategy: Unsorted Simple; Log Level: WARN.
- Configuration File:** last.conf
- Output:** LinkBrowser Log
- Close Up View:** shows a table with columns 'relates with' and 'to', and a 'show inferred relations' checkbox.
- Source Tree:** Lists classes: Address :ClazzImpl, Customer :ClazzImpl, Order :ClazzImpl, String :PrimitiveDataTypeImpl, address :AssociationImpl, customer :AssociationImpl.
- Status:** initialize source ready.

Right Screenshot: Output and Diagram

- System Linkbrowser:** Shows configuration for Source Domain (jmi_adapter_classdiagrams_offline.jar), Target Domain (jmi_adapter_dbschemata_offline.jar), and Link Domain (integration_classdiagrams_dbschemata.jar).
- Configuration File:** CDOfflineDSOffline.conf
- Action:** Algorithm: Forward Translation (Batch, Simple); Strategy: Unsorted Simple; Log Level: WARN.
- Configuration File:** last.conf
- Output:** LinkBrowser Log
- Close Up View:** shows a table with columns 'relates with' and 'to', and a 'show inferred relations' checkbox.
- Source Tree:** Lists classes: Address :ClazzImpl, Customer :ClazzImpl, Order :ClazzImpl, String :PrimitiveDataTypeImpl, address :AssociationImpl, customer :AssociationImpl, int :PrimitiveDataTypeImpl.
- Diagram:** A class diagram showing relationships between classes and their database constraints. Constraints are labeled: Address: Labeling, String: Columning, Customer: Labeling, name: Columning, Surname: Columning, Order: Labeling, id: Columning.
- Status:** perform operation ready.



Other Software Engineering Applications of Model Synchronization

- ▶ Mapping a PIM to a PSM in Model-Driven Architecture
- ▶ Graph Structurings (see course ST-II)
- ▶ Refactorings (see Course DPF)
- ▶ Semantic refinements
- ▶ Round-Trip Engineering (RTE)

The End: What Have We Learned

- ▶ Graph rewrite systems are tools to transform graph-based models and graph-based program representations
- ▶ MOFLON supports OCL queries and constraints
- ▶ TGG enable to bidirectionally map models and synchronize them
- ▶ Why can a TGG also be called a *metamodel mapping grammar*?