Conceptual Model Interoperability: a Metamodel-driven Approach

Pablo Rubén Fillottrani[1,2] and C. Maria Keet[3]

[1] Departamento de Ciencias e Ingeniería de la Computación, Universidad Nacional del Sur, Bahía Blanca, Argentina, prf@cs.uns.edu.ar
[2] Comisión de Investigaciones Científicas, Provincia de Buenos Aires, Argentina
[3] Department of Computer Science, University of Cape Town, South Africa, mkeet@cs.uct.ac.za

RuleML'14, Prague, August 18-20, 2014

Outline

Outline

Introduction

- Need for sharing data across information systems
- Interoperability at the level of conceptual models is a key
- Linking, converting, and integrating conceptual models represented in different modelling languages
- E.g.: database is designed with EER, the application layer that uses the database is specified in UML, and the business rules in ORM

Related works

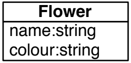
- One-off unidirectional algorithms to transform a language; e.g., ORM→UML [Bollen, 2002]
- Multi-language approaches,
  - linking each model to a graph [Boyd & McBrien, 2005]
  - description logic language unifier [Calvanese et al, 1999, Keet, 2012]
  - transformations mediated by a dictionary of common terms [Atzeni et al, 2012], or metamodel [Venable & Grundy, 1995]
- Problems: only partial solutions:
  - omit several constructs (e.g., weak entity types, roles) or modify the language (e.g., by removing datatypes from UML)
  - imprecise 'equivalence' mappings, or
  - the algorithms are not available
- Overall, there is very limited interoperability of conceptual data models in praxis

Outline

Ingredients

- Assert a link between two entities in different models and evaluate *automatically* whether it is a valid assertion and what it does entail
- Need to know what type of entities they are, whether they are the same, and if not, whether one can be transformed into the other for that particular selection.
- First step: how to transform that entity from one model into another
- Second: validate inter-model assertion

Overview transforming entities



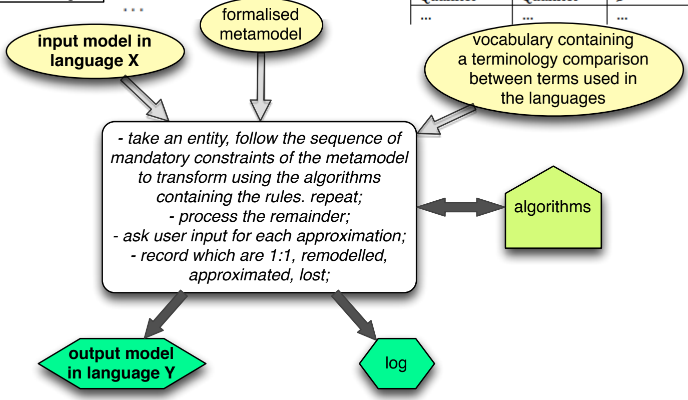
$\forall(x)(\text{Relationship}(x) \rightarrow \text{Entity}(x))$

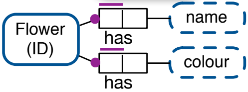
...

$\forall(x)(\neg(\text{Datatype}(x) \wedge \text{Qualifier}(x)))$

...

Metamodel	UML	EER	ORM2
...	...	...	...
Relationship	Association	Relationship	Fact type
...	...	...	...
Qualifier	Qualifier	ø	ø
...	...	...	...





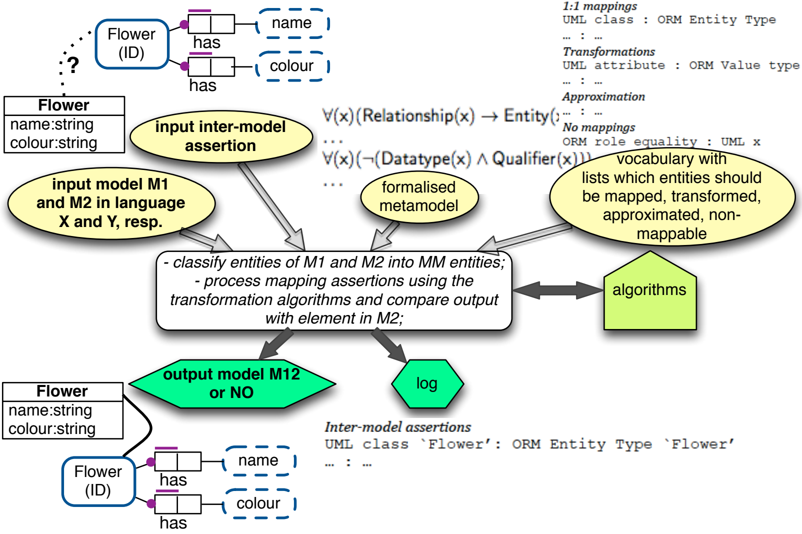
1:1 mappings

```
UML class 'Flower' -> ORM Entity Type 'Flower'
```

Transformations

```
UML attribute 'colour' -> ORM Value Type 'colour'
...
```

Overview validating inter-model assertions



Outline

1. Motivation
2. Approach
3. Rules
4. Validating mappings
5. Discussion and Conclusions

Design

- Rules between languages vs. 'through' the metamodel
- Metamodel-mediated:
  - reduces amount of mappings
  - extensibility
  - maintainability
  - use the constraints in the metamodel to induce firing the rules
- Static structural components:
  [Keet & Fillottrani, 2013a, Keet & Fillottrani, 2013b]

1:1 mapping rules and the metamodel (selection)

(R1) Association $\xrightarrow{\text{UML to MM}}$ Relationship

in:

`Association(AssociationEnd : Class, AssociationEnd : Class)`

out: `AssociationEnd` → Role *// i.e., using (Ro1)*

out: `Association` → Relationship

out: `Class` → Object Type *// i.e., using (O1)*

out: Relationship(Role:Object type, Role:Object Type)

(1R) Relationship $\xrightarrow{\text{MM to UML}}$ `Association`

in: Relationship(Role:Object type, Role:Object Type)

out: Role → `AssociationEnd` *// i.e., using (1Ro)*

out: Relationship → `Association`

out: Object Type → `Class` *// i.e., using (1O)*

out:

`Association(AssociationEnd : Class, AssociationEnd : Class)`

(xRx) Likewise for the other 1:1 mappings of Fact type and Relationship, with (1R) $\xrightarrow{\text{MM o UML}}$; (R2) $\xrightarrow{\text{ORM to MM}}$; (2R) $\xrightarrow{\text{MM to ORM}}$; (R3) $\xrightarrow{\text{EER to MM}}$; (3R) $\xrightarrow{\text{MM to EER}}$.

Generating and mapping

GenOT Class $\xrightarrow{\text{UML to ORM}}$ Entity type

in: $\mathtt{C}$

out: (O1)

out: (2O) // *i.e., an ORM* $\mathtt{EntityType}$ *named* $\mathtt{C}$

MapR Association $\xrightarrow{\text{UML to ER}}$ Relationship

in: $\mathtt{A}(\mathtt{ae}_1 : \mathtt{C}_1, \mathtt{ae}_2 : \mathtt{C}_2)$

out: (R1)

out: (3R)

out: match pattern out(3R) with $\mathtt{R}(\mathtt{rc}_1 : \mathtt{E}_1, \mathtt{rc}_2 : \mathtt{E}_2)$

Transformations (selection)

(VT) Value type

(V1) Value type $\xrightarrow{\text{ORM to MM}}$ Value type
in: `ValueType` ∧ `mapped_to(ValueType, DataType)`
out: (D1)
out: `mapped_to` → mapped_to
out: `ValueType` → Value type
out: ValueType ∧ mapped_to(Value type, Data type)

(1V) Value type $\xrightarrow{\text{ORM to MM}}$ Value type
.... *// steps in (V1) in reverse order*

(Att-VT) Attribute and Value type conversions

(VT-to-Att) Value type $\xrightarrow{\text{MM}}$ Attribute
in: Value type ∧ mapped_to(Value type, Data type)
out: (D1)
out: Object type
out: ValueType → Attribute
out: Attribute(Object type, Data type)

Mapping (selection)

MapVTAtt Value type $\xrightarrow{\text{ORM to UML}}$ Attribute

in: $\mathtt{V} \wedge \mathtt{mapped\_to}(\mathtt{V}, \mathtt{D})$

out: (V1)

out: (VT-to-Att)

out: (1A) *// i.e., a UML Class Diagram with* $\mathtt{A}(\mathtt{C}, \mathtt{D})$

out: match pattern out(1A) with attribute declaration in the UML diagram

Approximations (selection)

(Att) Attribute

(Ae1) Attribute $\rightsquigarrow_{\text{EER to MM}}$ Attribute
in: `Attribute(Class, __)`
out: (O1)
out: __ → *choose a* DataType
out: `Attribute` → Attribute
out: Attribute(Object type, Data type)

(1Ae) Attribute $\rightsquigarrow_{\text{MM to EER}}$ Attribute
in: Attribute(Object type, Data type)
out: (O1)
out: Attribute → `Attribute`
out: DataType → __
out: `Attribute(Class, __)`

Mapping

MapSID ORM reference scheme $\rightsquigarrow_{\text{ORM to EER}}$ EER single attribute identifier

in:

$\mathtt{FT}(\mathtt{r_e} : \mathtt{E_1}, \mathtt{r_v} : \mathtt{V}) \wedge \mathtt{mapped\_to}(\mathtt{V}, \mathtt{D}) \wedge \mathtt{M} \wedge \mathtt{C}(\mathtt{mic} = 1, \mathtt{mac} = 1)$

out: (O2) *// ORM entity type into MM object type*

out: (V1) *// ORM value type into MM value type*

out: (M2) *// ORM mandatory into MM mandatory*

out: (C2) *// ORM cardinality into MM cardinality*

out: (VT-to-Att) *// MM conversion value type to attribute*

out: (3O) *// MM object type into entity type* $\mathtt{E}$ *of EER*

out: (1Ae) *// generate EER Diagram attribute:* $\mathtt{A}(\mathtt{E}, \_\_)$

out: (3M) *// MM mandatory into mandatory of EER*

out: (3C) *// MM cardinality into cardinality of EER*

out: match pattern out(1Ae,3M,3C) with single identifier declaration in the EER diagram

Outline

Example: UML, ORM, relationships

- Example: an inter-model assertion between a UML binary association R_1 and an ORM fact type R_2
- Classify the entities in term of the metamodel entities
- Consider the 1:1 mappings, transformations, approximations, non-mappable entities.
- Then choose a direction for mapping validation, and the rules and formalised metamodel

5-step procedure

Step 1 Vocabulary: association and fact type correspond to Relationship in the metamodel, and thus enjoy a 1:1 mapping. Ruleset: R1 from UML to the metamodel and 2R to OMR's fact type.

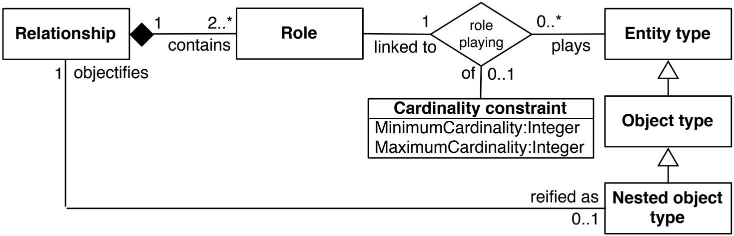
Step 2 First 'knock-on' effects: R1 and 2R refer to Role and Object type of the metamodel.
Metamodel states that there must be at least 2 contains relations from Relationship to Role.
Cause the role-rules to be evaluated, with Ro1 of R_1's two association ends and 2Ro for ORM's roles

5-step procedure

Step 1 Vocabulary: association and fact type correspond to Relationship in the metamodel, and thus enjoy a 1:1 mapping. Ruleset: R1 from UML to the metamodel and 2R to OMR's fact type.

Step 2 First 'knock-on' effects: R1 and 2R refer to Role and Object type of the metamodel.
Metamodel states that there must be at least 2 contains relations from Relationship to Role.
Cause the role-rules to be evaluated, with Ro1 of R_1's two association ends and 2Ro for ORM's roles

Small section of the metamodel, graphically



Formalised metamodel (section), highlighted for step 2

$\forall(x,y)(\texttt{Contains}(x,y) \rightarrow \texttt{Relationship}(x) \wedge \texttt{Role}(y))$

$\forall(x)\exists^{\geq 2}y(\texttt{Contains}(x,y))$

$\forall(x)(\texttt{Role}(x) \rightarrow \exists(y)(\texttt{Contains}(y,x)))$

$\forall(x,y,z)(\texttt{Contains}(x,y) \wedge \texttt{Contains}(z,y) \rightarrow (x = z))$

$\forall(x,y,z)(\texttt{RolePlaying}(x,y,z) \rightarrow \texttt{Role}(x) \wedge \texttt{CardinalityConstraint}(y) \wedge \texttt{EntityType}(z))$

$\forall(x)(\texttt{Role}(x) \rightarrow \exists(y,z)(\texttt{RolePlaying}(x,y,z)))$

$\forall(x,y,z,v,w)(\texttt{RolePlaying}(x,y,z) \wedge \texttt{RolePlaying}(x,v,w) \rightarrow (y = v) \wedge (z = w))$

$\forall(x,y,z,v,w)(\texttt{RolePlaying}(x,y,z) \wedge \texttt{RolePlaying}(v,y,w) \rightarrow (x = v) \wedge (z = w))$

$\forall(x)(\texttt{CardinalityConstraint}(x) \rightarrow \exists(y)(\texttt{MinimumCardinality}(x,y) \wedge \texttt{Integer}(y)))$

$\forall(x)(\texttt{CardinalityConstraint}(x) \rightarrow \exists(y)(\texttt{MaximumCardinality}(x,y) \wedge \texttt{Integer}(y)))$

$\forall(x,y)(\texttt{Identifies}(x,y) \rightarrow (\texttt{IdentificationConstraint}(x) \wedge \texttt{ObjectType}(y)))$

$\forall(x)(\texttt{IdentificationConstraint}(x) \rightarrow \exists(y)(\texttt{Identifies}(x,y)))$

$\forall(x,y,z)((\texttt{Identifies}(x,y) \wedge \texttt{Identifies}(x,z)) \rightarrow (y = z))$

$\forall(x)(\texttt{ObjectType}(x) \rightarrow \exists(y)(\texttt{Identifies}(y,x)))$

$\forall(x,y,z)((\texttt{DeclaredOn}(x,y) \wedge \texttt{DeclaredOn}(x,z) \wedge \texttt{IdentificationConstraint}(x) \wedge (\neg(y =$

$\quad (\texttt{ValueProperty}(y) \leftrightarrow \neg\texttt{AttributiveProperty}(z)))$

$\forall(x)(\texttt{IdentificationConstraint}(x) \rightarrow \exists(y)(\texttt{DeclaredOn}(x,y)))$

$\forall(x,y)((\texttt{DeclaredOn}(x,y) \wedge \texttt{SingleIdentification}(x)) \rightarrow (\texttt{Attribute}(y) \vee \texttt{ValueType}(y))$

$\forall(x)(\texttt{SingleIdentification}(x) \rightarrow \exists(y)(\texttt{DeclaredOn}(x,y))$

$\forall(x,y,z)((\texttt{SingleIdentification}(x) \wedge \texttt{DeclaredOn}(x,y) \wedge \texttt{DeclaredOn}(x,z)) \rightarrow (y = z))$

cont'd

Step 3 Metamodel: Role must participate in the relationship rolePlaying, and it has a participating Object type and optionally a Cardinality constraint.
Also 1:1 mappings

Step 4 The class participating in R_1 causes its rules to be evaluated, being an O1 to Object type and 2O to ORM's entity type.

cont'd

Step 3 Metamodel: Role must participate in the relationship rolePlaying, and it has a participating Object type and optionally a Cardinality constraint.
Also 1:1 mappings

Step 4 The class participating in R_1 causes its rules to be evaluated, being an O1 to Object type and 2O to ORM's entity type.

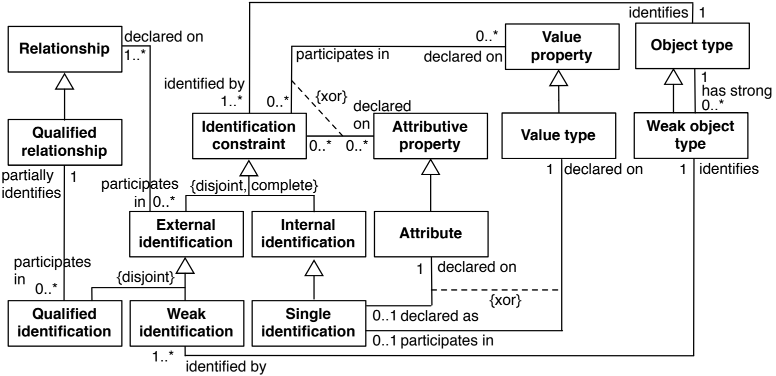
Highlighted section for step 3

$\forall(x, y)(\texttt{Contains}(x, y) \rightarrow \texttt{Relationship}(x) \wedge \texttt{Role}(y))$
$\forall(x)\exists^{\geq 2} y(\texttt{Contains}(x, y))$
$\forall(x)(\texttt{Role}(x) \rightarrow \exists(y)(\texttt{Contains}(y, x)))$
$\forall(x, y, z)(\texttt{Contains}(x, y) \wedge \texttt{Contains}(z, y) \rightarrow (x = z))$
$\forall(x, y, z)(\texttt{RolePlaying}(x, y, z) \rightarrow \texttt{Role}(x) \wedge \texttt{CardinalityConstraint}(y) \wedge \texttt{EntityType}(z))$
$\forall(x)(\texttt{Role}(x) \rightarrow \exists(y, z)(\texttt{RolePlaying}(x, y, z)))$
$\forall(x, y, z, v, w)(\texttt{RolePlaying}(x, y, z) \wedge \texttt{RolePlaying}(x, v, w) \rightarrow (y = v) \wedge (z = w))$
$\forall(x, y, z, v, w)(\texttt{RolePlaying}(x, y, z) \wedge \texttt{RolePlaying}(v, y, w) \rightarrow (x = v) \wedge (z = w))$
$\forall(x)(\texttt{CardinalityConstraint}(x) \rightarrow \exists(y)(\texttt{MinimumCardinality}(x, y) \wedge \texttt{Integer}(y)))$
$\forall(x)(\texttt{CardinalityConstraint}(x) \rightarrow \exists(y)(\texttt{MaximumCardinality}(x, y) \wedge \texttt{Integer}(y)))$
$\forall(x, y)(\texttt{Identifies}(x, y) \rightarrow (\texttt{IdentificationConstraint}(x) \wedge \texttt{ObjectType}(y)))$
$\forall(x)(\texttt{IdentificationConstraint}(x) \rightarrow \exists(y)(\texttt{Identifies}(x, y)))$
$\forall(x, y, z)((\texttt{Identifies}(x, y) \wedge \texttt{Identifies}(x, z)) \rightarrow (y = z))$
$\forall(x)(\texttt{ObjectType}(x) \rightarrow \exists(y)(\texttt{Identifies}(y, x)))$
$\forall(x, y, z)((\texttt{DeclaredOn}(x, y) \wedge \texttt{DeclaredOn}(x, z) \wedge \texttt{IdentificationConstraint}(x) \wedge (\neg(y =$
$(\texttt{ValueProperty}(y) \leftrightarrow \neg\texttt{AttributiveProperty}(z)))$
$\forall(x)(\texttt{IdentificationConstraint}(x) \rightarrow \exists(y)(\texttt{DeclaredOn}(x, y)))$
$\forall(x, y)((\texttt{DeclaredOn}(x, y) \wedge \texttt{SingleIdentification}(x)) \rightarrow (\texttt{Attribute}(y) \vee \texttt{ValueType}(y))$
$\forall(x)(\texttt{SingleIdentification}(x) \rightarrow \exists(y)(\texttt{DeclaredOn}(x, y)))$
$\forall(x, y, z)((\texttt{SingleIdentification}(x) \wedge \texttt{DeclaredOn}(x, y) \wedge \texttt{DeclaredOn}(x, z)) \rightarrow (y = z))$

cont'd

Step 5 Each Object type must have at least one Identification constraint.
and involving one or more attributes or value types.
If it is a Single identification, then a rule similar to MapSID is called and executed (which, in turn, calls the Att-to-VT rule and the use of Data type)

Small section of the metamodel, graphically



* A *Weak identification* is a combination of one or more *Attributive property* of the *Weak object type* it *identifies* together with the *Identification constraint* of the *Object type* it has a *Relationship* with and this *Object type* is disjoint with the *Weak object type*.

* The *Single identification* has a *Mandatory* constraint on the participating *Role* and the *Relationship* that *Role* is *contained in* has a 1:1 *Cardinality constraint* declared on it.

* *Qualified identification* and *External identification* are *declared on* only *Attributive property*.

* A *Qualified relationship participates in* a *Qualified identification* only if the *Cardinality constraint* is 1.

Formalised metamodel (section), highlighted for step 5

$$\forall(x,y)(\mathtt{Contains}(x,y) \rightarrow \mathtt{Relationship}(x) \land \mathtt{Role}(y))$$
$$\forall(x)\exists^{\geq 2}y(\mathtt{Contains}(x,y))$$
$$\forall(x)(\mathtt{Role}(x) \rightarrow \exists(y)(\mathtt{Contains}(y,x)))$$
$$\forall(x,y,z)(\mathtt{Contains}(x,y) \land \mathtt{Contains}(z,y) \rightarrow (x = z))$$
$$\forall(x,y,z)(\mathtt{RolePlaying}(x,y,z) \rightarrow \mathtt{Role}(x) \land \mathtt{CardinalityConstraint}(y) \land \mathtt{EntityType}(z))$$
$$\forall(x)(\mathtt{Role}(x) \rightarrow \exists(y,z)(\mathtt{RolePlaying}(x,y,z)))$$
$$\forall(x,y,z,v,w)(\mathtt{RolePlaying}(x,y,z) \land \mathtt{RolePlaying}(x,v,w) \rightarrow (y = v) \land (z = w))$$
$$\forall(x,y,z,v,w)(\mathtt{RolePlaying}(x,y,z) \land \mathtt{RolePlaying}(v,y,w) \rightarrow (x = v) \land (z = w))$$
$$\forall(x)(\mathtt{CardinalityConstraint}(x) \rightarrow \exists(y)(\mathtt{MinimumCardinality}(x,y) \land \mathtt{Integer}(y)))$$
$$\forall(x)(\mathtt{CardinalityConstraint}(x) \rightarrow \exists(y)(\mathtt{MaximumCardinality}(x,y) \land \mathtt{Integer}(y)))$$
$$\forall(x,y)(\mathtt{Identifies}(x,y) \rightarrow (\mathtt{IdentificationConstraint}(x) \land \mathtt{ObjectType}(y)))$$
$$\forall(x)(\mathtt{IdentificationConstraint}(x) \rightarrow \exists(y)(\mathtt{Identifies}(x,y)))$$
$$\forall(x,y,z)((\mathtt{Identifies}(x,y) \land \mathtt{Identifies}(x,z)) \rightarrow (y = z))$$
$$\forall(x)(\mathtt{ObjectType}(x) \rightarrow \exists(y)(\mathtt{Identifies}(y,x)))$$
$$\forall(x,y,z)((\mathtt{DeclaredOn}(x,y) \land \mathtt{DeclaredOn}(x,z) \land \mathtt{IdentificationConstraint}(x) \land (\neg(y =$$
$$(\mathtt{ValueProperty}(y) \leftrightarrow \neg\mathtt{AttributiveProperty}(z)))$$
$$\forall(x)(\mathtt{IdentificationConstraint}(x) \rightarrow \exists(y)(\mathtt{DeclaredOn}(x,y)))$$
$$\forall(x,y)((\mathtt{DeclaredOn}(x,y) \land \mathtt{SingleIdentification}(x)) \rightarrow (\mathtt{Attribute}(y) \lor \mathtt{ValueType}(y))$$
$$\forall(x)(\mathtt{SingleIdentification}(x) \rightarrow \exists(y)(\mathtt{DeclaredOn}(x,y))$$
$$\forall(x,y,z)((\mathtt{SingleIdentification}(x) \land \mathtt{DeclaredOn}(x,y) \land \mathtt{DeclaredOn}(x,z)) \rightarrow (y = z))$$

Outline

Considerations

- Upfront 'investment', notably in designing and formalising the metamodel
- Extra work pays off:
  - increased coverage of features
  - higher precision of mappings
  - approximations are explicit
  - coordination of rules thanks to constraints in metamodel (cf. plain dictionary)
- Rules usable for both transformations and validation
- Yet to be implemented and evaluated with actual models

Conclusions

- Metamodel-driven approach for model transformations and inter-model assertions where the models are *represented in different languages*: static structural, components of ER, EER, UML v2.4.1, ORM, and ORM2
- Uses formalised metamodel to direct a sequence of the language transformations
- Set of mapping, transformation, and approximation rules to carry it out
- Transformations (conversions) and validation of inter-model mappings

References I

- Atzeni, P., Gianforme, G., Cappellari, P.: Data model descriptions and translation signatures in a multi-model framework. Annals of Mathematics and Artificial Intelligence 63, 1–29 (2012)
- Bollen, P.W.L.: A formal ORM-to-UML mapping algorithm (2002), `http://arno.unimaas.nl/show.cgi?fid=46`, research memo RM 02/016, Faculty of Economics and Business Administration, University of Maastricht
- Boyd, M., McBrien, P.: Comparing and transforming between data models via an intermediate hypergraph data model. J. on Data Semantics IV, 69–109 (2005)
- Calvanese, D., Lenzerini, M., Nardi, D.: Unifying class-based representation formalisms. Journal of Artificial Intelligence Research 11, 199–240 (1999)
- Keet, C.M.: Ontology-driven formal conceptual data modeling for biological data analysis. In: Elloumi, M., Zomaya, A.Y. (eds.) Biological Knowledge Discovery Handbook: Preprocessing, Mining and Postprocessing of Biological Data, chap. 6, pp. 129–154. Wiley (2013)
- Keet, C.M., Fillottrani, P.R.: Structural entities of an ontology-driven unifying metamodel for UML, EER, and ORM2. In: Proc. of MEDI'13. LNCS, vol. 8216, pp. 188–199. Springer (2013), sept. 25-27, 2013, Amantea, Calabria, Italy
- Keet, C.M., Fillottrani, P.R.: Toward an ontology-driven unifying metamodel for UML class diagrams, EER, and ORM2. In: Proc. of ER'13. LNCS, vol. 8217, pp. 313–326. Springer (2013), 11-13 Nov., 2013, Hong Kong
- Venable, J., Grundy, J.: Integrating and supporting Entity Relationship and Object Role Models. In: Proc. of ER'95. LNCS, vol. 1021, pp. 318–328. Springer (1995)

Thank you!

Questions?