Unifying industry-grade class-based conceptual data modeling languages with $\mathcal{CM}_{com}$

C. Maria Keet

Faculty of Computer Science, Free University of Bozen-Bolzano, Italy
keet@inf.unibz.it

Abstract. From the side of modelers and early-adopter industry, interest in reasoning over conceptual models and other online usage of conceptual models is growing. To obtain a more precise insight in the characteristics of the main conceptual modeling languages, we define the (semi-)standardized ORM, ORM2, UML, ER, and EER diagram languages in terms of the new generic conceptual data modeling language $\mathcal{CM}_{com}$ that is based on the DL language $\mathcal{DLR}_{ifd}$. $\mathcal{CM}_{com}$ has the most expressive common denominator with these languages. $\mathcal{CM}_{com}$ advances prospects for automated, online, interoperability among diverse conceptual data models and ensures compatibility with and between industry-grade conceptual data modeling languages.

1 Introduction

The main commonly used industry-grade class-based conceptual data modelling languages are ER, EER, UML class diagrams, ORM, and ORM2 that come in various, mostly graphical, notational variants that share a common core, but also cater for specifics with the type of applications in mind. Several diagram-based transformations between these languages have been made where for each new notation—e.g., Barker ER, IE, and IDEF1X [21]—a new mapping scheme was identified [18], resulting in an m:n mesh with $(k-1)k/2$ mappings among k languages. A different approach that avoids this problem has been taken by [8, 10, 11] who aim to unify class-based modeling languages through the $\mathcal{DLR}$ family of Description Logic languages, focusing on information integration but also defining precise model-theoretic semantics for conceptual modeling languages that enables for logic-based 1:n mappings based on the language constructs irrespective of the variations in graphical elements. However, this has been worked out for restricted versions of ER, UML, and frame-based systems only, but not for full EER, UML class diagrams, and ORM/ORM2. In addition, in the meantime more expressive flavours of $\mathcal{DLR}$ have been investigated and experimented with to enable computational support for both integration of conceptual data models and automated satisfiability and consistency checking [13, 14]. We extend and refine this "common core" [11] by, first, integrating previously obtained results of advances on and mappings between conceptual modelling languages and do take into account standardized (UML, IDEF1X) and semi-standardized (Barker ER, IE, ORM, ORM2) conceptual data modeling languages and their implementations in modeling tools such as VisioModeler, NORMA, CaseTalk, RationalRose, VP-UML, and SmartDraw [1–26]. We identify their greatest common denominator, $\mathcal{DLR}_{ifd}$, which will be used to formally define the generic common conceptual data modeling language $\mathcal{CM}_{com}$ that thus has an equi-satisfiable $\mathcal{DLR}_{ifd}$ knowledge base. Second, this $\mathcal{CM}_{com}$ is used to define ER, EER, UML class diagrams, ORM, and ORM2, where it will be shown in an unambiguous way that UML, ER, EER, and ORM are different fragments of ORM2. $\mathcal{CM}_{com}$ thereby simplifies computational implementations for interoperability of conceptual data models modeled in different graphical languages; hence, being compatible with established modeling languages and practices yet moving forward toward realization of a logic-based, computer-aided, and conceptual modeling-based information integration framework.

In the remainder of the paper, we first deal with the $\mathcal{DLR}$ family and introduce the $\mathcal{CM}_{com}$ syntax and semantics (section 2). Subsequently, ER, EER, UML class diagrams, ORM, and ORM2 are defined in terms of $\mathcal{CM}_{com}$ in section 3. We conclude in section 4.

2 The $\mathcal{DLR}$ family of languages and $\mathcal{CM}_{com}$

For formal conceptual data modelling, we introduce $\mathcal{DLR}$ first [5], which was developed for providing a formal characterization of conceptual modelling languages to enable automated reasoning over the conceptual data models to improve their quality, and to use it as unifying paradigm for database integration through integrating their respective conceptual models [8, 10]. Take atomic relations ($\mathbf{P}$) and atomic concepts A as the basic elements of $\mathcal{DLR}$, which allows us to construct arbitrary relations (arity ≥ 2) and arbitrary concepts according to the following syntax:

$\mathbf{R} \longrightarrow \top_n | \ \mathbf{P} \ | \ (\$i/n : C) \ | \ \neg\mathbf{R} \ | \ \mathbf{R}_1 \sqcap \ \mathbf{R}_2$

$C \longrightarrow \top_1 | \ A \ | \ \neg C \ | \ C_1 \sqcap C_2 \ | \ \exists[\$i]\mathbf{R} \ | \ \leq k[\$i]\mathbf{R}$

i denotes a component of a relation; if components are not named, then integer numbers between 1 and n_{max} are used, where n is the arity of the relation. k is a nonnegative integer for cardinality constriants. Only relations of the same arity can be combined to form expressions of type $\mathbf{R}_1 \sqcap \ \mathbf{R}_2$, and $i \leq n$. The model-theoretic semantics of $\mathcal{DLR}$ is specified through the usual notion of interpretation, where $\mathcal{I}= (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, and the interpretation function $\cdot^{\mathcal{I}}$ assigns to each concept C a subset $C^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$ and to each n-ary $\mathbf{R}$ a subset $\mathbf{R}^{\mathcal{I}}$ of $(\Delta^{\mathcal{I}})^n$, such that the conditions are satisfied following Table 1. A *knowledge base* is a finite set $\mathcal{KB}$ of $\mathcal{DLR}$ (resp. $\mathcal{DLR}_{ifd}$) axioms of the form $C_1 \sqsubseteq C_2$ and $R_1 \sqsubseteq R_2$, and with R_1 and R_2 being relations of the same arity. An interpretation $\mathcal{I}$ satisfies $C_1 \sqsubseteq C_2$ ($R_1 \sqsubseteq R_2$) if and only if the interpretation of C_1 (R_1) is included in the interpretation of C_2 (R_2), i.e. $C_1^{\mathcal{I}(t)} \subseteq C_2^{\mathcal{I}(t)}$ ($R_1^{\mathcal{I}(t)} \subseteq R_2^{\mathcal{I}(t)}$). $\top_1$ denotes the interpretation domain, $\top_n$ for $n \geq 1$ denotes a subset of the n-cartesian product of the domain, which covers all introduced n-ary relations; hence "$\neg$" on rela-

Table 1. Semantics of $\mathcal{DLR}$ and $\mathcal{DLR}_{\mathit{ifd}}$.

$\top_n^{\mathcal{I}} \subseteq (\Delta^{\mathcal{I}})^n$	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
$\mathbf{P}^{\mathcal{I}} \subseteq \top_n^{\mathcal{I}}$	$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
$(\neg \mathbf{R})^{\mathcal{I}} = \top_n^{\mathcal{I}} \setminus \mathbf{R}^{\mathcal{I}}$	$(C_1 \sqcap C_2)^{\mathcal{I}} = C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
$(\mathbf{R}_1 \sqcap \mathbf{R}_2)^{\mathcal{I}} = \mathbf{R}_1^{\mathcal{I}} \cap \mathbf{R}_2^{\mathcal{I}}$	$(\$i/n : C)^{\mathcal{I}} = \{(d_1, ..., d_n) \in \top_n^{\mathcal{I}} \mid d_i \in C^{\mathcal{I}}\}$
$\top_1^{\mathcal{I}} = \Delta^{\mathcal{I}}$	$(\exists[\$i]\mathbf{R})^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \exists (d_1, ..., d_n) \in \mathbf{R}^{\mathcal{I}}.d_i = d\}$
	$(\leq k[\$i]\mathbf{R})^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \sharp\{(d_1, ..., d_n) \in \mathbf{R}_1^{\mathcal{I}} \mid d_i = d\} \leq k\}$

tions means difference rather than the complement. The $(\$i/n : C)$ denotes all tuples in $\top_n$ that have an instance of C as their i-th component.

There are four extensions to $\mathcal{DLR}$. The most interesting in the current scope is $\mathcal{DLR}_{\mathit{ifd}}$ [6], because it can represent most or all of the expressivity of common conceptual modelling languages. $\mathcal{DLR}_{\mathit{ifd}}$ supports $i$dentification assertions on a concept C, which has the form $(\mathbf{id}\ C[i_1]R_1, ..., [i_h]R_h)$, where each R_j is a relation and each i_j denotes one component of R_j. This construct allows for representation of external uniqueness in ORM, ER's weak entity types, and objectification. $\mathcal{DLR}_{\mathit{ifd}}$ also supports non-unary $f$unctional $d$ependency assertions on a relation R, which has the form $(\mathbf{fd}\ R\ i_1, ..., i_h \rightarrow j)$, where $h \geq 2$, and $i_1, ..., i_h, j$ denote components of R (unary **fd**s lead to undecidability [6]), which are useful for UML class diagram methods and ORM's derived-and-stored fact types. $\mathcal{DLR}_{\mu}$ contains the fixpoint construct for recursive structures over single-inheritance trees of a role [7] and thereby can represent acyclicity, transitivity, asymmetry, and (ir)reflexivity. $\mathcal{DLR}_{\mathit{reg}}$ includes *reg*ular expressions, which includes the role composition operator and reflexive transitive closure [9]. [2, 3] developed a temporal version ($\mathcal{DLR}_{\mathcal{US}}$), which has additional $\mathcal{U}$ntil and $\mathcal{S}$ince operators for temporal conceptual modelling with $\mathcal{ER}_{VT}$.

2.1 The generic common conceptual data model $\mathcal{CM}_{com}$

The formalisation adopted here is based on previous presentations [1, 6, 11] extended with EXTK and FD for $\mathcal{DLR}_{\mathit{ifd}}$'s **id** and **fd**, respectively, and making explicit objectification OBJ, subroles ISA_U, role exclusion REX, and disjunctive mandatory roles RDM; that is, given a particular conceptual data model in the generic conceptual data modeling language $\mathcal{CM}_{com}$, then there is an equi-satisfiable $\mathcal{DLR}_{\mathit{ifd}}$ knowledge base. The "new" REX and RDM (in $\mathcal{DLR}_{\mathit{ifd}}$ notation: $[r_i]R_i \sqsubseteq \neg[r_j]R_j$ and $C_i \sqsubseteq \sqcup_{i=1}^{n} \exists[r_j]R_i$ among n relations each for the jth role with $j \leq n$, respectively) have not been used other than for the ORM2 to $\mathcal{DLR}_{\mathit{ifd}}$ mapping [23, 24] because ORM and ORM2 do have these fine-grained notions, whereas UML, ER, and EER do not; given that they are not harmful at all to UML and (E)ER, they are added to $\mathcal{CM}_{com}$. We first introduce the syntax, illustrate it with an example, and then proceed to the semantics.

Definition 1. (Conceptual Data Model $\mathcal{CM}_{com}$ syntax) *A $\mathcal{CM}_{com}$ conceptual data model is a tuple*

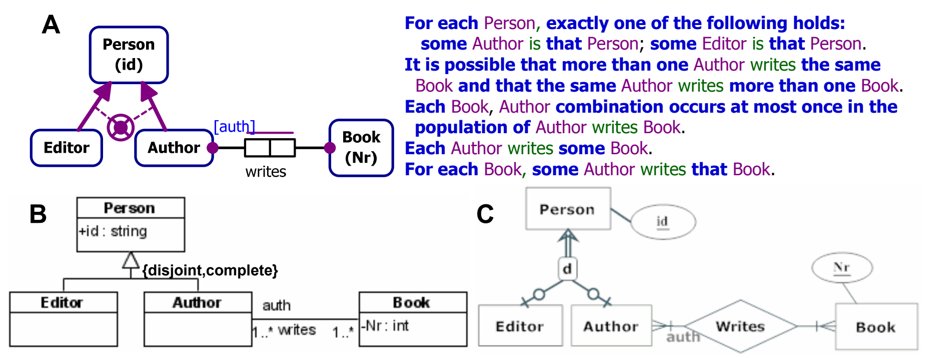
$\Sigma = (\mathcal{L}, \text{REL}, \text{ATT}, \text{CARD}, \text{ISA}_C, \text{ISA}_R, \text{ISA}_U, \text{DISJ}, \text{COVER}, \text{KEY}, \text{EXTK}, \text{FD}, \text{OBJ}$

$\textsc{rex}, \textsc{rdm})$
such that:

1. *$\mathcal{L}$ is a finite alphabet partitioned into the sets: $\mathcal{C}$ (*class *symbols), $\mathcal{A}$ (*attribute *symbols), $\mathcal{R}$ (*relationship *symbols), $\mathcal{U}$ (*role *symbols), and $\mathcal{D}$ (*domain *symbols); the tuple $(\mathcal{C}, \mathcal{A}, \mathcal{R}, \mathcal{U}, \mathcal{D})$ is the* signature *of the conceptual model Σ.*
2. *$\textsc{att}$ is a function that maps a class symbol in $\mathcal{C}$ to an $\mathcal{A}$-labeled tuple over $\mathcal{D}$, $\textsc{att}(C) = \langle A_1 : D_1, \ldots, A_h : D_h\rangle$.*
3. *$\textsc{rel}$ is a function that maps a relationship symbol in $\mathcal{R}$ to an $\mathcal{U}$-labeled tuple over $\mathcal{C}$, $\textsc{rel}(R) = \langle U_1 : C_1, \ldots, U_k : C_k\rangle$, and k is the* arity *of R.*
4. *$\textsc{card}$ is a function $\textsc{card} : \mathcal{C} \times \mathcal{R} \times \mathcal{U} \mapsto \mathbb{N} \times (\mathbb{N} \cup \{\infty\})$ denoting cardinality constraints. We denote with $\textsc{cmin}(C, R, U)$ and $\textsc{cmax}(C, R, U)$ the first and second component of $\textsc{card}$.*
5. *$\textsc{isa}_C$ is a binary relationship $\textsc{isa}_C \subseteq \mathcal{C} \times \mathcal{C}$.*
6. *$\textsc{isa}_R$ is a binary relationship $\textsc{isa}_R \subseteq \mathcal{R} \times \mathcal{R}$. $\textsc{isa}_R$ between relationships is restricted to relationships with the same arity.*
7. *$\textsc{isa}_U$ is a binary relationship $\textsc{isa}_U \subseteq \mathcal{U} \times \mathcal{U}$. $\textsc{isa}_U$ between roles of relationships is restricted to relationships with the same arity.*
8. *$\textsc{disj}, \textsc{cover}$ are binary relations over $2^{\mathcal{C}} \times \mathcal{C}$, describing disjointness and covering partitions, respectively, over a group of $\textsc{isa}$ that share the same superclass.*
9. *$\textsc{key}$ is a function, $\textsc{key} : \mathcal{C} \mapsto \mathcal{A}_i$, that maps a class symbol in $\mathcal{C}$ to its key attribute(s) where $1 < i \leq n$ and n denotes the arity of the relation or total amount of attributes of $\mathcal{C}$.*
10. *$\textsc{extk}$ is an identification assertion (external uniqueness / weak entity type) $\textsc{extk} : \mathcal{C} \mapsto \mathcal{R}_i \times \mathcal{U}_j$ where $1 < i \leq n$ and each R_i denotes a relation and each U_j denotes the component of R_i, with $1 \leq j \leq m$ where m denotes the arity of relation R_i (intuitively, such an assertion states that no two instances of C agree on the participation to $R_1, \ldots, R_i$).*
11. *$\textsc{fd}$ is a functional dependency assertion on a relation, $\textsc{fd} : \mathcal{R} \mapsto \mathcal{U}_i \times j$ where $i \geq 2$ and $U_1, ..., U_i, j$ denote components of R.*
12. *$\textsc{obj}$ is an objectification function that maps an n-ary relation symbol $R \in \mathcal{R}$ to n binary relations $r_1, \ldots r_n$ over $\mathcal{C}$ $(R' \in \mathcal{C})$, $\textsc{obj}(R) = \langle [U_1]r_1, \textsc{cmax}(R', r_1, U_1) = 1, \textsc{cmax}(C_1, r_1, U_2) = 1, \ldots, [U_1]r_n, \textsc{cmax}(R', r_n, U_1) = 1, \textsc{cmax}(C_n, r_n, U_2) = 1\rangle$ and $\textsc{extk}(R') = \langle U_1[r_1], \ldots, [U_n]r_n\rangle$.*
13. *$\textsc{rex}$, $\textsc{rdm}$ are binary relations over $2^{\mathcal{U}} \times \mathcal{U}$, describing disjointness partitions over a group of roles $\mathcal{U}$ of relations in $\mathcal{R}$ of the same arity to which $\mathcal{C}$ participates.*

The example below gives a flavour of how the $\mathcal{CM}_{com}$ syntax can map to icons in a graphical $\mathcal{CM}_{com}$ conceptual model that uses UML class diagram, EER, or ORM2 notation. In principle, one can map the $\mathcal{CM}_{com}$ syntax to any set of icons or fixed syntax (pseudo-) natural language as long as the relation between the $\mathcal{CM}_{com}$ syntax and icons or pseudo-NL has been specified.

Example: graphical and textual syntax for $\mathcal{CM}_{com}$. Mappings from $\mathcal{CM}_{com}$ syntax to UML, EER, and ORM2 are shown in Fig.1, where we have, among others, the ISA visualized with a directed arrow (e.g. `Author` ISA `Person`, in $\mathcal{DLR}_{ifd}$: `Author` $\sqsubseteq$ `Person`), cardinality constrains, such as CARD(`Author`, `Writes`, `auth`) $= (1, n)$ (in $\mathcal{DLR}_{ifd}$: `Author` $\sqsubseteq \exists$`[auth]writes`) with a "`1..*`" in UML, craw's feet and line in EER, and blob and line in ORM2, and DISJ and COVER with `disjoint, complete` in UML, in EER with double directed arrows and encircled `d`, and in ORM2 with an encircled blob with an X (e.g., `Author`, `Editor` are disjoint and cover `Person`, in $\mathcal{DLR}_{ifd}$: `Person` $\sqsubseteq$ `Author` $\sqcup$ `Editor` and `Author` $\sqsubseteq \neg$`Editor`). The text in bold in Fig.1-A is the fixed-syntax pseudo-natural language verbalizing the constraints in the ORM2 diagram. $\diamondsuit$



For each Person, **exactly one of the following holds:**
some Author **is that** Person; **some** Editor **is that** Person.
It is possible that more than one Author writes **the same** Book **and that the same** Author writes **more than one** Book.
Each Book, Author **combination occurs at most once in the population of** Author writes Book.
Each Author writes **some** Book.
For each Book, **some** Author writes **that** Book.

Fig. 1. Examples with ORM2 drawn and verbalized in NORMA (A), UML class diagram drawn in VP-UML (B), and EER drawn with SmartDraw (C).

The model-theoretic semantics associated with this generic $\mathcal{CM}_{com}$ modeling language is defined as follows.

Definition 2. **($\mathcal{CM}_{com}$ Semantics)** *Let Σ be a $\mathcal{CM}_{com}$ conceptual data model. An* interpretation *for the conceptual model Σ is a tuple $\mathcal{B} = (\Delta^{\mathcal{B}} \cup \Delta^{\mathcal{B}}_{D}, \cdot^{\mathcal{B}})$, such that:*

- *$\Delta^{\mathcal{B}}$ is a nonempty set of abstract objects disjoint from $\Delta^{\mathcal{B}}_{D}$;*
- *$\Delta^{\mathcal{B}}_{D} = \bigcup_{D_i \in \mathcal{D}} \Delta^{\mathcal{B}}_{D_i}$ is the set of basic domain values used in Σ; and*
- *$\cdot^{\mathcal{B}}$ is a function that maps:*
  - *Every basic domain symbol $D \in \mathcal{D}$ into a set $D^{\mathcal{B}} = \Delta^{\mathcal{B}}_{D_i}$.*
  - *Every class $C \in \mathcal{C}$ to a set $C^{\mathcal{B}} \subseteq \Delta^{\mathcal{B}}$—thus* objects *are instances of classes.*
  - *Every relationship $R \in \mathcal{R}$ to a set $R^{\mathcal{B}}$ of $\mathcal{U}$-labeled tuples over $\Delta^{\mathcal{B}}$—i.e. let R be an n-ary relationship connecting the classes $C_1, \ldots, C_n$, $\text{REL}(R) = \langle U_1 : C_1, \ldots, U_n : C_n \rangle$, then, $r \in R^{\mathcal{B}} \rightarrow (r = \langle U_1 : o_1, \ldots, U_n : o_n \rangle \wedge \forall i \in \{1, \ldots, n\}.o_i \in C^{\mathcal{B}}_i)$. We adopt the convention: $\langle U_1 : o_1, \ldots, U_n : o_n \rangle \equiv \langle o_1, \ldots, o_n \rangle$, when $\mathcal{U}$-labels are clear from the context.*

- *Every attribute* $A \in \mathcal{A}$ *to a set* $A^{\mathcal{B}} \subseteq \Delta^{\mathcal{B}} \times \Delta_D^{\mathcal{B}}$*, such that, for each* $C \in \mathcal{C}$*, if* $\textsc{att}(C) = \langle A_1 : D_1, \ldots, A_h : D_h \rangle$*, then,* $o \in C^{\mathcal{B}} \rightarrow (\forall i \in \{1, \ldots, h\}, \exists a_i.\ \langle o, a_i \rangle \in A_i^{\mathcal{B}} \land \forall a_i.\langle o, a_i \rangle \in A_i^{\mathcal{B}} \rightarrow a_i \in \Delta_{D_i}^{\mathcal{B}})$.

$\mathcal{B}$ *is said a* legal database state *or* legal application software state *if it satisfies all of the constraints expressed in the conceptual data model:*

– *For each* $C_1, C_2 \in \mathcal{C}$*: if* $C_1 \ \textsc{isa}_C \ C_2$*, then* $C_1^{\mathcal{B}} \subseteq C_2^{\mathcal{B}}$.
– *For each* $R_1, R_2 \in \mathcal{R}$*: if* $R_1 \ \textsc{isa}_R \ R_2$*, then* $R_1^{\mathcal{B}} \subseteq R_2^{\mathcal{B}}$.
– *For each* $U_1, U_2 \in \mathcal{U}$*,* $R_1, R_2 \in \mathcal{R}$*,* $\textsc{rel}(R_1) = \langle U_1 : o_1, \ldots, U_n : o_n \rangle$*,* $\textsc{rel}(R_2) = \langle U_2 : o_2, \ldots, U_m : o_m \rangle$*,* $n = m$*,* $R_1 \neq R_2$*: if* $U_1 \ \textsc{isa}_U \ U_2$*, then* $U_1^{\mathcal{B}} \subseteq U_2^{\mathcal{B}}$.
– *For each* $R \in \mathcal{R}$ *with* $\textsc{rel}(R) = \langle U_1 : C_1, \ldots, U_k : C_k \rangle$*: all instances of* R *are of the form* $\langle U_1 : o_1, \ldots, U_k : o_k \rangle$ *where* $o_i \in C_i^{\mathcal{B}}$ *and* $1 \leq i \leq k$.
– *For each cardinality constraint* $\textsc{card}(C, R, U)$*, then:*
$o \in C^{\mathcal{B}} \rightarrow \textsc{cmin}(C, R, U) \leq \#\{r \in R^{\mathcal{B}} \mid r[U] = o\} \leq \textsc{cmax}(C, R, U)$.
– *For all* $C, C_1, \ldots, C_n \in \mathcal{C}$*: if* $\{C_1, \ldots, C_n\} \ \textsc{disj} \ C$*, then,*
$\forall i \in \{1, \ldots, n\}.C_i \ \textsc{isa} \ C \land \forall j \in \{1, \ldots, n\},\ j \neq i.C_i^{\mathcal{B}} \cap C_j^{\mathcal{B}} = \emptyset$.
– *For all* $C, C_1, \ldots, C_n \in \mathcal{C}$*: if* $\{C_1, \ldots, C_n\} \ \textsc{cover} \ C$*, then,*
$\forall i \in \{1, \ldots, n\}.C_i \ \textsc{isa} \ C \land C^{\mathcal{B}} = \bigcup_{i=1}^{n} C_i^{\mathcal{B}}$.
– *For each* $C \in \mathcal{C}, A \in \mathcal{A}$ *such that* $\textsc{key}(C) = A$*, then* A *is an attribute and* $\forall a \in \Delta_D^{\mathcal{B}}.\#\{o \in C^{\mathcal{B}} \mid \langle o, a \rangle \in A^{\mathcal{B}}\} \leq 1$.
– *For each* $C \in \mathcal{C}$*,* $R_h \in \mathcal{R}$*,* $h \geq 1$*,* $\textsc{rel}(R_h) = \langle U : C, U_1 : C_1, \ldots, U_k : C_k \rangle$*,* $k \geq 1$*,* $k+1$ *the arity of* R_h*, such that* $\textsc{extk}(C) = [U_1]R_1, \ldots, [U_h]R_h$*, then for all* $o_a, o_b \in C^{\mathcal{B}}$ *and for all* $t_1, s_1 \in R_1^{\mathcal{B}}$*, ...,* $t_h, s_h \in R_h^{\mathcal{B}}$ *we have that:*

$$\left.\begin{aligned} o_a = t_1[U_1] = ... = t_h[U_h] \\ o_b = s_1[U_1] = ... = s_h[U_h] \\ t_j[U] = s_j[U],\ \textit{for } j \in \{1, ..., h\},\ \textit{and for } U \neq j \end{aligned}\right\} \quad \textit{implies } o_a = o_b$$

where o_a *is an instance of* C *that is the* U_j*-th component of a tuple* t_j *of* R_j*, for* $j \in \{1, ..., h\}$*, and* o_b *is an instance of* C *that is the* U_j*-th component of a tuple* s_j *of* R_j*, for* $j \in \{1, ..., h\}$*, and for each* j*,* t_j *agrees with* s_j *in all components different from* U_j*, then* o_a *and* o_b *are the same object.*
– *For each* $R \in \mathcal{R}$*,* $U_i, j \in \mathcal{U}$*, for* $i \geq 2$*,* $i \neq j$*,* $\textsc{rel}(R) = \langle U_1 : C_1, \ldots, U_i : C_i, j : C_j \rangle$*,* $\textsc{fd}(R) = \langle U_1, \ldots, U_i \rightarrow j \rangle$*, then for all* $t, s \in R^{\mathcal{B}}$*, we have that* $t[U_1] = s[U_1], ..., t[U_i] = s[U_i]$ *implies* $t_j = s_j$.
– *For each* $R, r_1, \ldots, r_n \in \mathcal{R}$*,* $R', C_1, \ldots, C_n \in \mathcal{C}$*,* $U_1, \ldots, U_n, u_s, u_t \in \mathcal{U}$*,* R *has arity* n*,* $\textsc{rel}$*,* $\textsc{cmax}$*, and* $\textsc{extk}$ *interpreted as above, such that* $\textsc{obj}(R) = \langle [u_s]r_1, \textsc{cmax}(R', r_1, u_s) = 1, \textsc{cmax}(C_1, r_1, u_t) = 1, \ldots, [u_s]r_n, \textsc{cmax}(R', r_n, u_s) = 1, \textsc{cmax}(C_n, r_n, u_t) = 1 \rangle$*, then*
$\forall i \in \{2, \ldots, n\}.\{U_i, u_s, u_t \in U^{\mathcal{B}} \land r_i \in R^{\mathcal{B}} \land o_i, r' \in C^{\mathcal{B}} \land \langle U_1 : o_1, U_2 : o_2, \ldots, U_n : o_n \rangle \in R^{\mathcal{B}} \mid r_i[u_s] = r' \land r_i[u_t] = o_i\}$.
– *For each* $U_i \in \mathcal{U}$*,* $i \geq 2$*,* $R_i \in \mathcal{R}$*, each* R_i *has the same arity* m *(with* $m \geq 2$*),* $C_j \in \mathcal{C}$ *with* $2 \leq j \leq i(m-1)+1$*, and* $\textsc{rel}(R_i) = \langle U_i : C_i, \ldots U_m : C_m \rangle$ *(and, thus,* $R_i \in R_i^{\mathcal{B}}$ *and* $o_j \in C_j^{\mathcal{B}}$*), if* $\{U_1, U_2, \ldots U_{i-1}\} \ \textsc{rex} \ U_i$*, then*
$\forall i \in \{1, \ldots, i\}.o_j \in C_j^{\mathcal{B}} \rightarrow \textsc{cmin}(o_j, r_i, u_i) \leq 1 \land u_i \neq u_1 \land \ldots \land u_i \neq u_{i-1}$
where $u_i \in U_i^{\mathcal{B}}$*,* $r_i \in R_i^{\mathcal{B}}$.

– *For each* $U_i \in \mathcal{U}$*,* $i \geq 2$*,* $R_i \in \mathcal{R}$*, each* R_i *has the same arity* m *(with* $m \geq 2$*),* $C_j \in \mathcal{C}$ *with* $2 \leq j \leq i(m-1)+1$*, and* $\text{REL}(R_i) = \langle U_i : C_i, \ldots U_m : C_m \rangle$*, if* $\{U_1, U_2, \ldots U_{i-1}\}$ RDM U_i*, then*
$\forall i \in \{1, \ldots, n\}.o_j \in C_j^{\mathcal{B}} \rightarrow \text{CMIN}(o_j, r_i, u_i) \geq 1$ *where* $u_i \in U_i^{\mathcal{B}}$*,* $r_i \in R_i^{\mathcal{B}}$*.*

3 $\mathcal{CM}_{com}$ for ER, EER, UML, ORM, and ORM2

Defining the considered languages in terms of $\mathcal{CM}_{com}$, one encounters two main problems: (i) establish what is, or is not, "the" (E)ER and ORM language, and if textual or OCL constraints count, and (ii) decide what to do with an officially informal language (UML [25]) or if there are alternative formalisations (ORM [17, 20]). If one has a formal definition of the syntax and semantics of a conceptual modeling language as with $\mathcal{CM}_{com}$, it is utterly unambiguous as to what is, and what is not, in the language. One approach to address this is to also provide the full syntax and semantics of ER, EER, UML class diagrams, ORM, and ORM2 in the same fashion as $\mathcal{CM}_{com}$. Alternatively, one could take industry-grade CASE tools and take the *de facto* standards, or a consensus where industry-grade tools, standards, prototypes, and research are converging. Given that the more distant goal of the comparison is to have, on the one hand, a common, formal, foundation for conceptual data modeling languages, yet on the other hand, interoperability among conceptual models in different graphical syntax—hence, permitting maintaining the diversity—we take the converging approach. An informal graphical rendering of the relations between the considered languages is depicted in Fig.2.

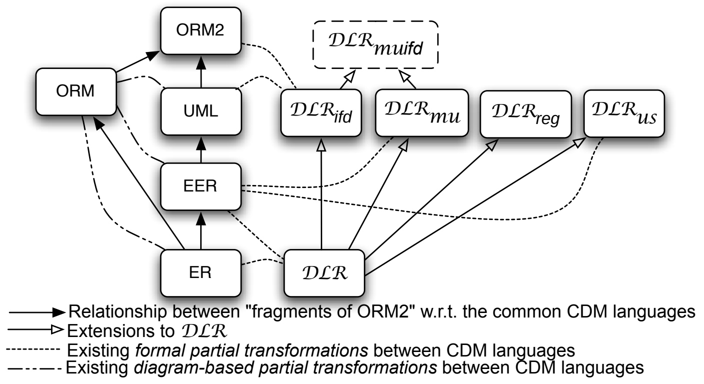

ORM2
$\mathcal{DLR}_{muifd}$
ORM
UML
$\mathcal{DLR}_{ifd}$
$\mathcal{DLR}_{mu}$
$\mathcal{DLR}_{reg}$
$\mathcal{DLR}_{us}$
EER
ER
$\mathcal{DLR}$
Relationship between "fragments of ORM2" w.r.t. the common CDM languages
Extensions to $\mathcal{DLR}$
Existing *formal partial transformations* between CDM languages
Existing *diagram-based partial transformations* between CDM languages

Fig. 2. Relations between common conceptual data modeling languages and the $\mathcal{DLR}$ family of Description Logic languages; $\mathcal{CM}_{com}$ has an equi-satisfiable $\mathcal{DLR}_{ifd}$ knowledge base. The higher up in the figure, the more constructors available in the language.

ER and EER. There is no "the" ER: Chen's original proposal for ER does include weak entity types [12] (EXTK), but this is not propagated to all ER tools and diagrammatic variants. In addition, most ER flavours neither have n-ary relations when $n > 2$ nor subtyping of entity types (e.g. Barker ER and some IE versions [18]), whereas IDEF1X (the US NIST standard 184) does include $\textsc{isa}_C$ [21]; these are all certainly in EER, as well as full CARD. Note that $\textsc{isa}_R$ is originally not specified in EER [15] and FD is available in relational schemas but not in (E)ER. We define ER and EER in terms of fragments of $\mathcal{CM}_{com}$ as follows.

Definition 3. ($\mathcal{CM}_{ER}$) *A $\mathcal{CM}_{ER}$ conceptual data model is a tuple*
$\Sigma = (\mathcal{L}, \textsc{rel}, \textsc{att}, \textsc{card}^{-}, \textsc{key}, \textsc{extk})$
adhering to $\mathcal{CM}_{com}$ syntax and semantics except that CARD *is restricted to any of the values $\{\geq 0, \leq 1, \geq 1\}$, denoted in Σ with* $\textsc{card}^{-}$.

Definition 4. ($\mathcal{CM}_{EER}$) *A $\mathcal{CM}_{EER}$ conceptual data model is a tuple*
$\Sigma = (\mathcal{L}, \textsc{rel}, \textsc{att}, \textsc{card}, \textsc{isa}_C, \textsc{disj}, \textsc{cover}, \textsc{key}, \textsc{extk})$
adhering to $\mathcal{CM}_{com}$ syntax and semantics.

UML class diagrams. The main problem with defining UML class diagrams is that UML officially lacks a formal semantics [25]. Berardi et al. [4] gave a formal semantics to UML class diagrams both in FOL and $\mathcal{DLR}_{ifd}$ but not in terms of a conceptual data modeling language and they do not address UML's OCL and shared and composite aggregation (roughly: part-of and proper-part-of, respectively). For both the FOL and $\mathcal{DLR}_{ifd}$ formalisation of aggregation, the semantics of the standard association relation (DL-role) is used; hence, all issues with representing part-whole relations (e.g., [16]) are ignored. This poses no real practical problem, because thanks to the lack of formal semantics of the UML specification, one can choose the semantics as one pleases (and most DL languages do not have a primitive for parthood relations anyway). The elusive "extra" is added as PW that can be characterized, at least, as a binary relation. Further, the *formal* UML and Icom [4, 13] have access to components of an association, but UML [25] and traditional CASE tools such as RationalRose and VP-UML do not, so that one cannot use $\textsc{isa}_U$ (hence, no REX and RDM either).

Definition 5. ($\mathcal{CM}_{UML}$) *A $\mathcal{CM}_{UML}$ conceptual data model is a tuple*
$\Sigma = (\mathcal{L}, \textsc{rel}, \textsc{att}, \textsc{card}, \textsc{isa}_C, \textsc{isa}_R, \textsc{disj}, \textsc{cover}, \textsc{key}, \textsc{extk}, \textsc{fd}, \textsc{obj}, \textsc{pw})$
adhering to $\mathcal{CM}_{com}$ syntax and semantics, except for the aggregation association PW*, with a syntax* $\textsc{pw} = \langle U_1 : C_1, U_2 : C_2\rangle$*, that has no defined semantics.*

Thus, *de facto*, $\mathcal{CM}_{UML} \subset \mathcal{CM}_{com}$ up until a proper, coherent syntax and semantics is defined for PW and full access to association ends is supported in either the UML specification or mainstream modeling tools, or both.

ORM and ORM2. ORM enjoys the comparative advantage of a having characterizations in FOL [17, 20], but variations have been implemented in tools, such as VisioModeler, NORMA, and CaseTalk, and not all extensions in ORM2 have been formalized [19]. Moreover, ORM is undecidable due to constraints with patterns of the type "constraint x over k ORM-roles" over an n-ary relation, $n \geq 3$, and $k < n$ because they correspond to arbitrary projections [23, 24]. On the flip side, an analysis of 168 ORM diagrams with about 1800 constraints made with LogicBlox software revealed that such constructs are rarely used in practice [26]. $\mathcal{CM}_{com}$ is a proper subset of ORM2, but, given Halpin's formalization [17], this is not the case for ORM. Due to space limitations, we do not repeat Halpin's formalization here, and omit role values (a preliminary addition to the language) and deontic constraints (requires deontic logic).

Definition 6. *A $\mathcal{CM}_{ORM}$ conceptual data model is a tuple*
$\Sigma = (\mathcal{L}, \text{REL}, \text{ATT}, \text{CARD}, \text{ISA}_C, \text{ISA}_R, \text{ISA}_U, \text{KEY}, \text{EXTK}, \text{FD}, \text{OBJ}, \text{REX}, \text{RDM}, \text{JOIN}, \text{KROL}, \text{RING}^-)$
adhering to $\mathcal{CM}_{com}$ syntax and semantics and, in addition:
- JOIN *comprises* {join-subset, join-equality, join-exclusion} *as defined in [17].*
- KROL *comprises* {subset over k roles, multi-role frequency, set-equality over k roles, role exclusion over k roles} *as defined in [17].*
- RING$^-$ *comprises* {intransitive, irreflexive, asymmetric}*, as defined in [17].*

Definition 7. *A $\mathcal{CM}_{ORM2}$ conceptual data model is a tuple*
$\Sigma = (\mathcal{L}, \text{REL}, \text{ATT}, \text{CARD}, \text{ISA}_C, \text{ISA}_R, \text{ISA}_U, \text{DISJ}, \text{COVER}, \text{KEY}, \text{EXTK}, \text{FD}, \text{OBJ}, \text{REX}, \text{RDM}, \text{JOIN}, \text{KROL}, \text{RING})$
adhering to $\mathcal{CM}_{com}$ syntax and semantics, , and:
- KROL *and* JOIN *are as in Definition 6.*
- RING *comprises* {intransitive, irreflexive, asymmetric, antisymmetric, acyclic, symmetric}*, as defined in [17, 18].*

The gap regarding RING can be met mostly with $\mathcal{DLR}_\mu$, but then one cannot represent EXTK, OBJ, and FD. Given that $\mathcal{DLR}_\mu$ requires and extended interpretation function with valuation ρ on $\mathcal{I}$ [7] but that adding **id** and **fd** does not add any concept or role constructors (but uses a generalised ABox and Skolem constants [6]), adding **id** and **fd** to $\mathcal{DLR}_\mu$ looks most promising, provided a relation participating in least/greatest fixpoint is binary and is not also used in an **id** assertion to ensure avoiding cycles, and the non-unary **fd** limitation from $\mathcal{DLR}_{ifd}$ carries over to a $\mathcal{DLR}_{\mu ifd}$. Further, two comparative aspects merit additional consideration. First, KEY is for *single attribute* keys—ORM2 reference scheme—whereas for a key consisting of multiple 'attributes', i.e., an n-ary ORM *fact type* where $n > 2$ and an internal uniqueness over > 1 role, or external uniqueness, one has to use EXTK. In addition, ORM requires that an internal uniqueness constraint over k roles in an n-ary relation has to span $\geq n - 1$ roles to have an *elementary fact type*. For instance, an entity type and relation as Course($\underline{\text{Course\_Name}}$, Uni_Year, Teacher) is valid in ER, EER, UML, and $\mathcal{CM}_{\mathcal{DLR}}$, but is not permitted in ORM and ORM2, whereas Course($\underline{\text{Course\_Name, Uni\_Year}}$, Teacher) is valid in ER, EER, ORM, and ORM2, but it is not in the $\mathcal{DLR}$ family except for $\mathcal{DLR}_{ifd}$ and, hence, $\mathcal{CM}_{com}$. Second, ORM and ORM2 are attribute-free languages whereas we have ATT in $\mathcal{CM}_{ORM}$

and $\mathcal{CM}_{ORM2}$; this can be dealt with easily and faithful to both ORM and the underlying $\mathcal{DLR}$s: a $\mathcal{CM}_{com}$ $\textsc{att}(C) = \langle A : D \rangle$ ends up in $\mathcal{DLR}_{\mathit{ifd}}$ just like in ORM: $C \sqsubseteq \exists[U_1]A \sqcap (\leq 1[U_1]A) \sqcap \forall[U_1](A \Rightarrow (U_2 : D))$.

From the extant definitions it has become immediately clear that standard UML in $\mathcal{CM}_{UML}$ is within ExpTime-complete complexity [5]—because $\mathcal{DLR}_{\mathit{ifd}}$ is—and ER and EER of lower complexity [1] because they have no subrelations. Overall, $\mathcal{DLR}_{\mathit{ifd}}$ provides the most expressive common denominator among the languages and thereby provides the, thus far, best trade-off between expressiveness for conceptual data modeling and computation. Although for conceptual modeling the emphasis is more often on the former, computation has the distinct advantage of automated satisfiability and consistency checking with reasoners such as Racer and Pellet or custom-made ones [22, 26] before generating a database or other application software. Moreover, the common formal foundation with $\mathcal{CM}_{com}$ offers an "interchange" to simplify integration of conceptual data models made in different conceptual data modeling languages. Also, each modeler could, in principle, continue using her preferred conceptual modeling language, yet have diagrams that are fully compatible with the other types, as well as have automated precise translations; hence, CASE tool developers can choose any variation of the diagrammatical rendering of the $\mathcal{CM}_{com}$ syntax and still avail of the transferrable results presented here. Thus, the $\mathcal{CM}_{com}$ approach comparison updates [11] with more recent research into the $\mathcal{DLR}$ family and industry-grade conceptual modeling languages and by availing of DL languages, $\mathcal{CM}_{com}$ augments the current informal mapping in e.g. [18], thereby moving closer to *online interoperability* between ORM/ORM2, UML, and EER through a formal correspondence between the conceptual modelling languages with the $\mathcal{DLR}$ family as unifying paradigm.

4 Conclusions

The main commonly used industry-grade conceptual data modelling languages—ER, EER, UML class diagrams, ORM, and ORM2—were formally defined by means of the unambiguously defined $\mathcal{CM}_{com}$ conceptual modeling language as greatest common denominator that also has an equi-satisfiable $\mathcal{DLR}_{\mathit{ifd}}$ knowledge base. It was demonstrated that UML, ER, EER, and ORM, are fragments of ORM2 and that $\mathcal{CM}_{com}$ simplifies interoperability of conceptual data models modeled in different graphical languages. We are currently examining if a "$\mathcal{DLR}_{\mu\mathit{ifd}}$" remains within the decidable fragment of FOL to address ORM's ring constraints and part-whole's relational properties.

References

1. Artale, A., Calvanese, D., Kontchakov, R., Ryzhikov, V., Zakharyaschev, M. Reasoning over Extended ER Models. *Proc. of ER'07*, Springer LNCS 4801, 277-292.
2. Artale, A., Parent, C. & Spaccapietra, S. Evolving objects in temporal information systems. *AMAI*, 2007, 50(1-2), 5-38.

3. Artale, A., Franconi, E., Wolter, F., Zakharyaschev, M. A temporal description logic for reasoning about conceptual schemas and queries. In: *Proc. of JELIA '02*, S. Flesca, S. Greco, N. Leone, G. Ianni (Eds.), Springer, 2002, LNAI 2424, 98-110.
4. Berardi, D., Calvanese, D., De Giacomo, G. Reasoning on UML class diagrams. *Artificial Intelligence*, 2005, 168(1-2):70-118.
5. Calvanese, D., De Giacomo, G. Expressive description logics. In: *The Description Logic Handbook*, Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (Eds). Cambridge University Press, 2003. 178-218.
6. Calvanese, D., De Giacomo, G., Lenzerini, M. Identification constraints and functional dependencies in Description Logics. In *Proc. of IJCAI 2001*, 2001, 155-160.
7. Calvanese, D., De Giacomo, G., Lenzerini, M. Reasoning in expressive description logics with fixpoints based on automata on infinite trees. In: *Proc. of IJCAI'99*, 84-89.
8. Calvanese, D., De Giacomo, G., Lenzerini, M., Nardi, D., Rosati, R. Description logic framework for information integration. In: *Proc. of KR'98*, 2-13.
9. Calvanese, C., De Giacomo, G., Lenzerini, M. On the decidability of query containment under constraints. In: *Proc. of PODS'98*, 149-158, 1998.
10. Calvanese, D., Lenzerini, M., Nardi, D. Description logics for conceptual data modeling. In J. Chomicki and G. Saake, editors, *Logics for Databases and Information Systems*. Kluwer, Amsterdam. 1998.
11. Calvanese, D., Lenzerini, M. & Nardi, D. Unifying class-based representation formalisms. *Journal of Artificial Intelligence Research*, 11:199-240, 1999.
12. Chen, P.P. The Entity-Relationship model—Toward a unified view of data. *ACM Transactions on Database Systems*, 1976, 1(1): 9-36.
13. Fillottrani, P., Franconi, E., Tessaris, S. The new ICOM ontology editor. In: *Proc. of DL 2006*, Lake District, UK. May 2006.
14. Franconi, E., Ng, G. The iCom tool for intelligent conceptual modelling. *7th Int'l Workshop on KRDB*. Berlin, Germany. 2000.
15. Gogolla, M., Hohenstein, U. Towards a semantic view of an Extended Entity-Relationship Model. *ACM Transactions on Database Systems*, 1991, 16(3), 369-416.
16. Guizzardi, G. *Ontological foundations for structural conceptual models*. PhD Thesis, Telematica Institute, Twente University, Enschede, the Netherlands. 2005.
17. Halpin, T.A. *A logical analysis of information systems: static aspects of the data-oriented perspective*. PhD Thesis, University of Queensland, Australia. 1989.
18. Halpin, T. *Information Modeling and Relational Databases*. San Francisco: Morgan Kaufmann Publishers, 2001.
19. Halpin, T. ORM2. In *Proc. of ORM 2005*. Cyprus, 3-4 November 2005. In: OTM Workshops 2005. Halpin, T., Meersman, R. (eds.), Springer LNCS 3762, 676-687.
20. Hofstede, A.H.M. ter, Proper, H.A.. How to Formalize It? Formalization Principles for Information Systems Development Methods. *Information and Software Technology*, 1998, (40(10): 519-540.
21. Integrated Definition Methods (IDEF1X). http://www.idef.com/IDEF1X.html.
22. Kaneiwa, K., Satoh, K. Consistency Checking Algorithms for Restricted UML Class Diagrams. In: *Proc. of FoIKS '06*, Springer LNCS 3861, 2006. 219-239.
23. Keet, C.M. Prospects for and issues with mapping the Object-Role Modeling language into $\mathcal{DLR}_{\mathsf{ifd}}$. In: *Proc. of DL 2007*. CEUR-WS Vol-250, 331-338.
24. Keet, C.M. *Mapping the Object-Role Modeling language ORM2 into Description Logic language* $\mathcal{DLR}_{\mathsf{ifd}}$. arXiv:cs.LO/0702089v1.
25. Object Management Group. *Unified Modeling Language: Superstructure*. v2.0. formal/05-07-04. http://www.omg.org/cgi-bin/doc?formal/05-07-04.
26. Smaragdakis, Y., Csallner, C., Subramanian, R. Scalable automatic test data generations from modeling diagrams. In: *ASE'07*, Nov. 5-9, Atlanta, Georgia, USA. 4-13.