CHAPTER 1

ONTOLOGY-DRIVEN FORMAL CONCEPTUAL DATA MODELING FOR BIOLOGICAL DATA ANALYSIS

Catharina Maria Keet

School of Computer Science, University of KwaZulu-Natal, South Africa

1.1 INTRODUCTION

Biological data modeling serves many purposes, and many approaches exist that are used in this endeavor. The main topics of advanced conceptual data modeling for database and Object-Oriented software development to support biological data analysis are included in Figure 1.1, which extend the traditional 'waterfall' software development methodology as depicted in bold in Figure 1.2. The scope of this chapter is to provide an overview of the ontological and logical aspects of conceptual data modeling tailored to molecular biology and biological knowledge discovery.

Many databases and software applications have been and are being developed in bioinformatics, which, following good computing methodologies, are—or should have been—developed in stages, going from requirements analysis ('what should the envisioned software do?') and conceptual analysis ('what data should it be able to manage?') to design-level code and then to the actual implementation. It is well-known that omitting the conceptual analysis stage by going straight to coding or scripting just adds to the pile of one-off (bioinformatics) tools that have more bugs and are much less, or not at all, maintainable and interoperable. Conversely, availing of a proper software development methodology with a represen-

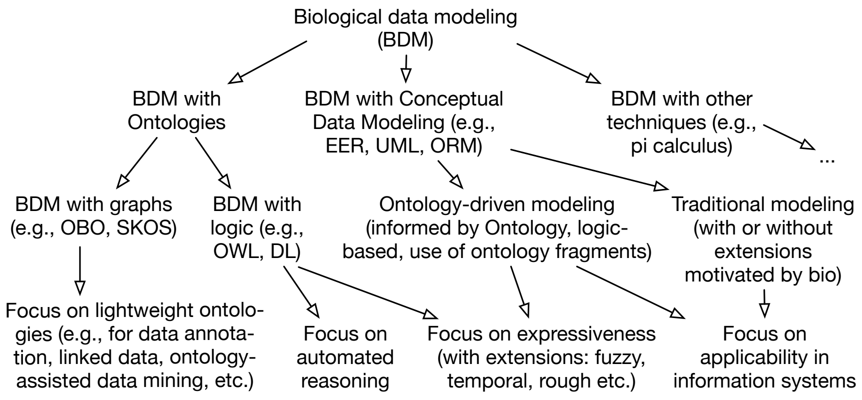

Biological data modeling (BDM)
BDM with Ontologies
BDM with Conceptual Data Modeling (e.g., EER, UML, ORM)
BDM with other techniques (e.g., pi calculus)
...
BDM with graphs (e.g., OBO, SKOS)
BDM with logic (e.g., OWL, DL)
Ontology-driven modeling (informed by Ontology, logic-based, use of ontology fragments)
Traditional modeling (with or without extensions motivated by bio)
Focus on lightweight ontologies (e.g., for data annotation, linked data, ontology-assisted data mining, etc.)
Focus on automated reasoning
Focus on expressiveness (with extensions: fuzzy, temporal, rough etc.)
Focus on applicability in information systems

Figure 1.1 Informal overview of the main subtopics in Biological Data Modeling with ontologies and conceptual data modeling and several usage scenarios.

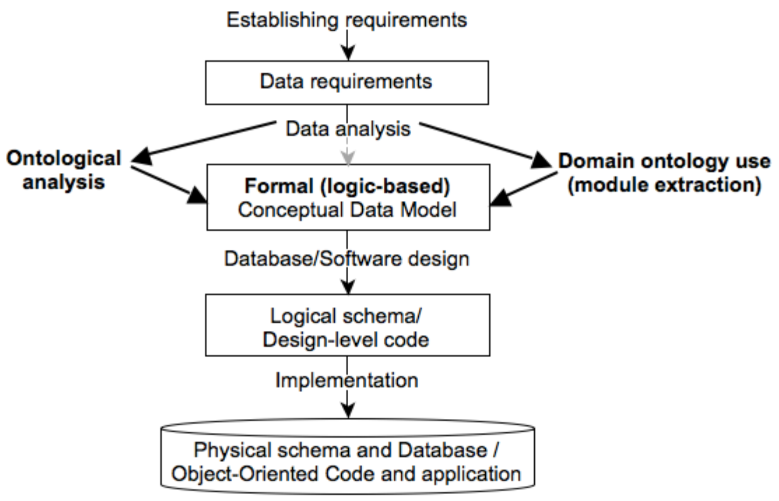

Establishing requirements
Data requirements
Data analysis
Ontological analysis
Domain ontology use (module extraction)
Formal (logic-based) Conceptual Data Model
Database/Software design
Logical schema/ Design-level code
Implementation
Physical schema and Database / Object-Oriented Code and application

Figure 1.2 The waterfall methodology, augmented with ontological analysis considering aspects from Ontology, Artificial Intelligence (knowledge representation), and ontologies, depicted in boldface.

tation at the conceptual layer has been shown to result in mitigation, avoidance, and/or solving such issues, and therewith contributed to a software infrastructure that enabled more sophisticated biological data analysis and knowledge discovery [15, 24, 48, 54, 58, 59, 63]. The output of the conceptual analysis stage for software development is a conceptual data model, normally represented in a language such as Extended Entity-Relationship (EER) for relational databases [18], the Unified Modeling Language (UML) for object-oriented software [52], or Object-Role Modeling (ORM) for either one [30], which have been used also for software development in bioinformatics; e.g., [11, 15, 21, 34, 54, 58]. These languages are not equivalent, one language may be better for biological data modeling than another

[34], and modelers tend to prefer one graphical language over another. So, assessing the expressiveness of those languages and the associated quality of the conceptual data models are of vital importance to ascertain their suitability for biological data modeling. This has been investigated as a component of *ontology-driven conceptual data modeling* to create ontology-driven information systems [25], where an ontology is positioned as an *application independent* formal representation of (our understanding of) a piece of reality and a conceptual data model as an *implementation independent* representation that is tailored to the application scenario. For instance, one can make more precise the representation of UML's aggregation association or part-whole relation by availing of advances in Ontology (philosophy) and making them applicable to conceptual data modeling [3, 29, 41]; one can then use scientific arguments and explain why, e.g., a protein chain's Residue's Coordinates are not part of the residue [11], but an attribute that describes its location, and distinguish between spatial containment and structural parthood [41]. Moreover, only if the data is stored and managed correctly can one achieve the most comprehensive and reliable discovery of biological knowledge. An ontology also can be used to generate multiple conceptual data models [20, 32, 61], which improves their quality and ensures interoperability between them. Both scenarios require a formal, logic-based, foundation of a conceptual data modeling language to foster precision, accuracy, adequate coverage of the subject domain semantics, and implementability. A benefit of the logic foundations is that a conceptual data model then can be subjected to automated reasoning services, such as consistency checking and deriving implicit constraints [1, 9, 16, 22, 38, 56], thereby improving its quality further and, hence, preventing software bugs. In addition, automated reasoning can be used as a tool for biological knowledge discovery [39, 63].

Thus, a necessary first step is to fix a formalization for the main conceptual data modeling languages, thereby differentiating between human-computer interaction issues and the real language expressiveness, hence clearing up part of the argument regarding suitability of a particular conceptual data modeling language for biology. Moreover, this enables not only assessing but also extending the real modeling features of the languages and laying the basis for applications of automated reasoning. The here proposed formalization into one formal common conceptual data modeling language—called $\mathcal{CM}_{com}$, based on the $\mathcal{DLR}_{\mathit{ifd}}$ Description Logic language—captures most of ORM and all of UML Class Diagram and EER language features. Conceptual data models are then improved upon with extensions coming from Ontology and ontologies in the form of modeling guidelines to improve the quality of conceptual data models, and extensions to the representation language, therewith solving certain outstanding modeling issues. This will be illustrated by markedly enriching the representation of, among others, catalytic reactions, transforming entities, and pathway information. Thanks to the formal foundation, automated reasoning services can be used conventionally and 'unconventionally' to find (derive) implicit knowledge and be used in *in silico* hypothesis testing, thereby contributing to biological knowledge discovery. To this end, a class classification and three main query scenarios will be introduced and illustrated.

The remainder of this chapter introduces $\mathcal{CM}_{com}$ (Section 1.2), ontological and language extensions to represent more complex knowledge more precise (Section 1.3), and how automated reasoning can benefit biological knowledge discovery (Section 1.4). We conclude in Section 1.5.

1.2 DESCRIPTION LOGICS FOR CONCEPTUAL DATA MODELING

Description Logics (DL) languages are decidable fragments of first order logic and are used for logic-based knowledge representation. They have been shown useful for reasoning both over conceptual models like ER and UML [6, 9, 16] and ontology languages such as OWL [50]. All DL languages have *concepts* (classes) and *roles* (relationships, n-ary predicates with $n \geq 2$), and have several constructs, therewith giving greater or lesser expressivity to the language and efficiency of reasoning over the logical theory. DL knowledge bases are composed of the *Terminological Box* (TBox), which contains axioms at the concept-level, and the *Assertional Box* (ABox) that contains assertions about instances; refer to [6] for details.

We first introduce $\mathcal{DLR}$ [12], which was developed to provide a formal characterization of conceptual data modeling languages to enable automated reasoning over them, to use it as unifying paradigm for database integration through integrating their respective conceptual data models [16], and to compare conceptual data modeling languages [36]. The basic elements of $\mathcal{DLR}$ are atomic relations ($\mathbf{P}$) and atomic concepts A, which allows construction of arbitrary relationships (arity ≥ 2) and concepts according to the following syntax:

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

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

where 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, and k is a nonnegative integer for cardinality constraints. 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 an *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 domain $\Delta^{\mathcal{I}}$ and assigns 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 1.1.

A *knowledge base* is a finite set $\mathcal{KB}$ of $\mathcal{DLR}$ (or $\mathcal{DLR}_{ifd}$) axioms of the form $C_1 \sqsubseteq C_2$ and $R_1 \sqsubseteq R_2$. 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. $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$ ($R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}}$). $\top_1$ 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

Table 1.1 Semantics of $\mathcal{DLR}$ and $\mathcal{DLR}_{ifd}$.

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

relations. The "($\$i/n : C$)" denotes all tuples in $\top_n$ that have an instance of C as their i-th component. The following abbreviations hold: $C_1 \sqcup C_2$ for $\neg(\neg C_1 \sqcap \neg C_2)$, $C_1 \Rightarrow C_2$ for $\neg C_1 \sqcup C_2$, $(\geq k[i]R)$ for $\neg(\leq k-1[i]R)$, $\exists[i]R$ for $(\geq 1[i]R)$, $\forall[i]R$ for $\neg\exists[i]\neg R$, $R_1 \sqcup R_2$ for $\neg(\neg R_1 \sqcap \neg R_2)$, and $(i : C)$ for $(i/n : C)$ when n is clear from the context. Note that a qualified role $\exists P.C$ is represented in $\mathcal{DLR}_{\mathit{ifd}}$ as $\exists[\$1](P \sqcap (\$2/2 : C))$, its inverse, $\exists P^{-}.C$, as $\exists[\$2](P \sqcap (\$1/2 : C))$, likewise for universal quantification ($\forall P.C$ as $\neg\exists[\$1](P \sqcap (\$2/2 : \neg C))$ and its inverse $\forall P^{-}.C$ as $\neg\exists[\$2](P \sqcap (\$1/2 : \neg C))$ [12].

There are four extensions to $\mathcal{DLR}$. The most relevant in the current scope are $\mathcal{DLR}_{\mathit{ifd}}$ [13], because it can capture most or all of the common conceptual modeling language features, and $\mathcal{DLR}_{\mathcal{US}}$, because it has an expressive temporal extension. $\mathcal{DLR}_{\mathit{ifd}}$ has two additional constructs compared to $\mathcal{DLR}$. It has $i$dentification assertions on a concept C, which have 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 is useful for external uniqueness in ORM, weak entity types in ER, and objectification. It also caters for non-unary $f$unctional $d$ependency assertions on a relationship R, which have 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, which are useful primarily for UML's methods and ORM's derived-and-stored fact types. Observe that there is no change in semantic rules because the algorithm for the extensions is checked against a (generalized) ABox [13]. $\mathcal{DLR}_{\mathcal{US}}$ has the $\mathcal{U}$ntil and $\mathcal{S}$ince operators for temporal ontologies and EER conceptual data models ($\mathcal{ER}_{VT}$) [5, 2], which has been used for modeling essential parthood [3], relation migration [42]. Generally, temporal extensions are very useful to represent constraints in molecular biology, such as formally characterizing that some enzymatic reaction happens only *after* another, *precedes* it, or happens *concurrently*.

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

Given the $\mathcal{DLR}_{\mathit{ifd}}$ syntax and semantics, we now can define the $\mathcal{CM}_{com}$ conceptual data modeling language; that is, given a particular conceptual data model in the generic conceptual data modeling language $\mathcal{CM}_{com}$, there is an equi-satisfiable $\mathcal{DLR}_{\mathit{ifd}}$ knowledge base. The formalization adopted here is based on previous presentations [1, 13, 17, 36], with two principal extensions so as make better use of the 'ifd' features of $\mathcal{DLR}_{\mathit{ifd}}$ that have not been addressed in those earlier works. These are the representation of weak entity types, UML's (hardly used) sub-association end and ORM's subroles, role exclusion, and disjunctive mandatory roles, which were hitherto used only for the more recent ORM2 to $\mathcal{DLR}_{\mathit{ifd}}$ mapping [38]. Given that they are not harmful at all to UML and (E)ER, and UML CASE tool features are moving in this direction, they are included in $\mathcal{CM}_{com}$. Also, refinements with respect to [36] are made concerning cardinality on relationships and on attributes, disjointness of classes and of relations, and more precise constraints are added on components of a relationship. We introduce the $\mathcal{CM}_{com}$ syntax in Definition 1, illustrate it with an example by mapping several elements of the syntax to graphical elements of EER, UML, and ORM2, proceed to the semantics in Definition 2, and then show the mapping from $\mathcal{CM}_{com}$ to $\mathcal{DLR}_{\mathit{ifd}}$ in Definition 3.

Definition 1 (Conceptual Data Model $\mathcal{CM}_{com}$ syntax) *A $\mathcal{CM}_{com}$ conceptual data model is a tuple* $\Sigma = (\mathcal{L}, \text{REL}, \text{ATT}, \text{CARD}_R, \text{CARD}_A, \text{ISA}_C, \text{ISA}_R, \text{ISA}_U, \text{DISJ}_C, \text{COVER}_C, \text{DISJ}_R, \text{KEY}, \text{EXTK}, \text{FD}, \text{OBJ}, \text{REX}, \text{RDM})$ *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. ATT *is a function that maps a class symbol in $\mathcal{C}$ to an $\mathcal{A}$-labeled tuple over $\mathcal{D}$,* $\text{ATT} : \mathcal{A} \mapsto \mathcal{D}$, *so that* $\text{ATT}(C) = \{A_1 : D_1, \ldots, A_h : D_h\}$ *where h a non-negative integer.*
3. REL *is a function that maps a relationship symbol in $\mathcal{R}$ to an $\mathcal{U}$-labeled tuple over $\mathcal{C}$,* $\text{REL}(R) = \{U_1 : C_1, \ldots, U_k : C_k\}$*, k is the* arity *of R, and if $(U_i, C_i) \in \text{REL}(R)$ then $\text{PLAYER}(R, U_i) = C_i$ and $\text{ROLE}(R, C_i) = U_i$. The signature of the relation is $\sigma_R = \langle \mathcal{U}, \mathcal{C}, \text{PLAYER}, \text{ROLE} \rangle$, where for all $U_i \in \mathcal{U}$, $C_i \in \mathcal{C}$, if $\sharp U \geq \sharp C$ then for each u_i, c_i, $\text{REL}(R)$, we have $\text{PLAYER}(R, U_i) = C_i$ and $\text{ROLE}(R, C_i) = U_i$, and if $\sharp U > \sharp C$ then $\text{PLAYER}(R, U_i) = C_i$, $\text{PLAYER}(R, U_{i+1}) = C_i$ and $\text{ROLE}(R, C_i) = U_i, U_{i+1}$.*
4. CARD_R *is a function* $\text{CARD}_R : \mathcal{C} \times \mathcal{R} \times \mathcal{U} \mapsto \mathbb{N} \times (\mathbb{N} \cup \{\infty\})$ *denoting cardinality constraints. We denote with $\text{CMIN}(C, R, U)$ and $\text{CMAX}(C, R, U)$ the first and second component of CARD_R.*
5. CARD_A *is a function* $\text{CARD}_A : \mathcal{C} \times \mathcal{A} \mapsto \mathbb{N} \times (\mathbb{N} \cup \{\infty\})$ *denoting multiplicity constraints for attributes. We denote with $\text{CMIN}(C, A)$ and $\text{CMAX}(C, A)$ the first and second component of CARD_A, and $\text{CARD}_A(C, A)$ may be defined only if $(A, D) \in \text{ATT}(C)$ for some $D \in \mathcal{D}$;*
6. ISA_C *is a binary relationship* $\text{ISA}_C \subseteq \mathcal{C} \times \mathcal{C}$.
7. ISA_R *is a binary relationship* $\text{ISA}_R \subseteq \mathcal{R} \times \mathcal{R}$. ISA_R *between relationships is restricted to relationships with the same signature, i.e., given an $R_1 \subseteq R_2$ then $\sigma_{R_1} = \sigma_{R_2}$, and $\text{PLAYER}(R_1, U_i) \subseteq \text{PLAYER}(R_2, U_i)$.*
8. ISA_U *is a binary relationship* $\text{ISA}_U \subseteq \mathcal{U} \times \mathcal{U}$. ISA_U *between roles of relationships is restricted to relationships with the same signature, i.e., given an $R_1 \subseteq R_2$ then $\sigma_{R_1} = \sigma_{R_2}$, and $\text{ROLE}(R_1, C_i) \subseteq \text{ROLE}(R_2, C_i)$.*
9. $\text{DISJ}_C, \text{COVER}_C$ *are binary relationships over $2^{\mathcal{C}} \times \mathcal{C}$, describing disjointness and covering partitions, respectively, over a group of* ISA *that share the same superclass.*
10. DISJ_R *is a binary relationship over $2^{\mathcal{R}} \times \mathcal{R}$, describing disjointness over a group of relations.*
11. KEY *is a function,* $\text{KEY} : \mathcal{C} \mapsto \mathcal{A}$*, that maps a class symbol in $\mathcal{C}$ to its key attribute and $A \in \mathcal{A}$ is an attribute already defined in $\text{ATT}(C)$, i.e., $\text{KEY}(C)$ may be defined only if $(A, D) \in \text{ATT}(C)$ for some $D \in \mathcal{D}$.*
12. EXTK *is called an identification assertion / external uniqueness / weak entity type, which is a function that maps a class to a set of relation-role pairs or attributes,* $\text{EXTK} : \mathcal{C} \mapsto 2^{2^{(\mathcal{R} \times \mathcal{U}) \cup \mathcal{A}}}$*, where $R \in \mathcal{R}$, $\text{REL}(R)$ so that $\text{PLAYER}(R, U) = C$ (with $C \in \mathcal{C}$), and for any participating $A \in \mathcal{A}$ such that $(A, D) \in \text{ATT}(C)$ for some $D \in \mathcal{D}$.*
13. FD *is a functional dependency assertion on a relation,* $\text{FD} : \mathcal{R} \mapsto 2^{2^{\mathcal{U}}} \times \mathcal{U}$ *where $U_1, ..., U_i, U_j \in \mathcal{U}$ denoting components of $R \in \mathcal{R}$; $\text{FD}(R)$ may be defined only if $\text{ROLE}(R, C_i) \in \text{REL}(R)$ and $i \geq 2$.*

14. OBJ *is an objectification function that maps an n-ary relation symbol $R \in \mathcal{R}$ to n binary relations $r_1, \ldots r_n \in \mathcal{R}$ over $\mathcal{C}$, i.e.,* OBJ : $\mathcal{R} \mapsto \mathcal{C}$. *Whenever* OBJ$(R) = R'$ *with* $R' \in \mathcal{C}$, ROLE$(R, C_i) = U_i$ *and, with* ROBJ *denoting the relationification of the role,* ROBJ$(U_i) = r_i$ *where r_i is a new binary relation,* REL$(r_i) = \{u_1 : R', u_2 : C_i\}$, $2 \leq i \leq n$, *and* EXTK$(R') = \{u_1[r_1], \ldots, u_1[r_n]\}$, *and* CMAX$(C, r_i, u_1) = 1$.
15. REX, 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.*

Thus, the explicit new features in $\mathcal{CM}_{com}$ compared to previous DL-focussed definitions of conceptual modeling languages are ISA$_U$, DISJ$_R$, EXTK, FD, OBJ, REX, and RDM, which thereby introduces also the distinction between 'simple' keys (KEY) and other keys like external uniqueness and natural keys (EXTK), and FDs for UML methods and ORM's derived and derived-and-stored fact types.

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-natural language has been specified. Put differently, with $\mathcal{CM}_{com}$ the mappings between the conceptual data modeling languages can be from one syntax (and semantics) to many graphical and textual representations instead of developing and maintaining m:n mappings between the various graphical languages. The following example demonstrates the principal mechanism for $\mathcal{CM}_{com}$ syntax and icons in UML Class Diagram, EER, and ORM2 notation.

Example 1 (Graphical syntaxes for $\mathcal{CM}_{com}$) Figure 1.3 depicts a UML, an EER, and an ORM2 diagram. The mappings from $\mathcal{CM}_{com}$ syntax to these graphics are:

`Author` ISA `Person` *(directed arrow in UML, EER, ORM2)*

CARD(`Author`, `Writes`, `auth`) $= (1, n)$
*(*1..* *in UML, craw's feet and line in EER, blob and line in ORM2)*

KEY(`Person`) = `id` *(underlined* id *in EER,* (id) *in ORM2)*

{`Author`, `Editor`} DISJ `Person`
({disjoint} *in UML, encircled* d *in EER, encircled* X *in ORM2)*

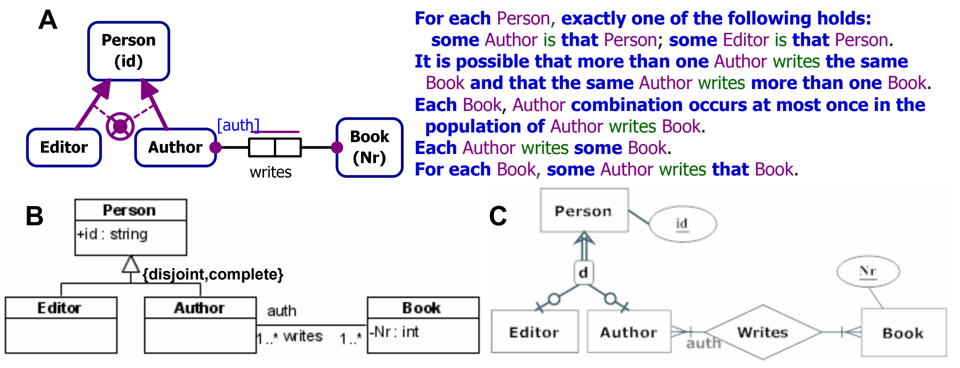
{`Author`, `Editor`} COVER `Person`
({complete} *in UML, open shaft arrow in EER, encircled blob in ORM2)*

Looking back at $\mathcal{DLR}_{ifd}$ in the introduction of this section and ahead to the demonstration that $\mathcal{CM}_{com}$ has an equi-satisfiable knowledge base (Definition 3), the equivalent representation in $\mathcal{DLR}_{ifd}$ is as follows:

$\texttt{Author} \sqsubseteq \texttt{Person}$ *(subsumption)*
$\texttt{Author} \sqsubseteq \exists[\texttt{auth}]\texttt{writes}$ *(at least one)*
$\texttt{Person} \sqsubseteq \exists^{=1}[\texttt{From}]\texttt{id}$, $\top \sqsubseteq \exists^{\leq 1}[\texttt{To}](\texttt{id} \sqcap [\texttt{From}] : \texttt{Person})$ *(key)*
$\texttt{Author} \sqsubseteq \neg\texttt{Editor}$ *(disjoint)*
$\texttt{Person} \sqsubseteq \texttt{Author} \sqcup \texttt{Editor}$ *(covering)*

One also can map the syntax to pseudo-natural language; e.g., for the ISAand NORMA's verbalization pattern *Each ... is an instance of ...* one obtains for `Author`ISA `Person` a domain expert readable surface rendering of *Each Author is an instance of Person*; the others are shown in Figure 1.3-A. ♢

The model-theoretic semantics associated with $\mathcal{CM}_{com}$ is as follows.



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.

Figure 1.3 Examples of graphical syntaxes for $\mathcal{CM}_{com}$ with an ORM2 diagram drawn in NORMA that also provides pseudo-natural language renderings (A), a UML Class Diagram drawn in VP-UML (B), and an EER diagram drawn with SmartDraw (C).

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 $\mathcal{I} = (\Delta^{\mathcal{I}} \cup \Delta^{\mathcal{I}}_D, \cdot^{\mathcal{I}})$, such that:*

- *$\Delta^{\mathcal{I}}$ is a nonempty set of abstract objects disjoint from $\Delta^{\mathcal{I}}_D$;*
- *$\Delta^{\mathcal{I}}_D = \bigcup_{D_i \in \mathcal{D}} \Delta^{\mathcal{I}}_{D_i}$ is the set of basic domain values used in Σ; and*
- *$\cdot^{\mathcal{I}}$ is a function that maps:*
  - *Every basic domain symbol $D \in \mathcal{D}$ into a set $D^{\mathcal{I}} = \Delta^{\mathcal{I}}_{D_i}$.*
  - *Every class $C \in \mathcal{C}$ to a set $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$.*
  - *Every relationship $R \in \mathcal{R}$ to a set $R^{\mathcal{I}}$ of $\mathcal{U}$-labeled tuples over $\Delta^{\mathcal{I}}$— i.e. let R be an n-ary relationship connecting the classes $C_1, \ldots, C_n$, $\text{REL}(R) = \{U_1 : C_1, \ldots, U_n : C_n\}$, then, $r \in R^{\mathcal{I}} \rightarrow (r = \{U_1 : o_1, \ldots, U_n : o_n\} \land \forall i \in \{1, \ldots, n\}.o_i \in C^{\mathcal{I}}_i)$.*
  - *Every attribute $A \in \mathcal{A}$ to a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}_D$, such that, for each $C \in \mathcal{C}$, if $\text{ATT}(C) = \{A_1 : D_1, \ldots, A_h : D_h\}$, then, $o \in C^{\mathcal{I}} \rightarrow (\forall i \in \{1, \ldots, h\}, \exists a_i.\ \langle o, a_i \rangle \in A^{\mathcal{I}}_i \land \forall a_i.\langle o, a_i \rangle \in A^{\mathcal{I}}_i \rightarrow a_i \in \Delta^{\mathcal{I}}_{D_i})$.*

$\mathcal{I}$ 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 $C_1 \text{ ISA}_C C_2$, then $C^{\mathcal{I}}_1 \subseteq C^{\mathcal{I}}_2$.*
- *For each $R_1, R_2 \in \mathcal{R}$: if $R_1 \text{ ISA}_R R_2$, then $R^{\mathcal{I}}_1 \subseteq R^{\mathcal{I}}_2$.*
- *For each $U_1, U_2 \in \mathcal{U}$, $R_1, R_2 \in \mathcal{R}$, $\text{REL}(R_1) = \{U_1 : o_1, \ldots, U_n : o_n\}$, $\text{REL}(R_2) = \{U_1 : o_1, \ldots, U_m : o_m\}$, $m = n$, $R_1 \neq R_2$: if $U_1 \text{ ISA}_U U_2$, then $U^{\mathcal{I}}_1 \subseteq U^{\mathcal{I}}_2$.*
- *For each $R \in \mathcal{R}$ with $\text{REL}(R) = \{U_1 : C_1, \ldots, U_k : C_k\}$: all instances of R are of the form $\{U_1 : o_1, \ldots, U_k : o_k\}$ where $o_i \in C^{\mathcal{I}}_i$, $U_i \in U^{\mathcal{I}}_i$, and $1 \leq i \leq k$.*
- *For each cardinality constraint $\text{CARD}_R(C, R, U)$, then:*
  $o \in C^{\mathcal{I}} \rightarrow \text{CMIN}(C, R, U) \leq \#\{r \in R^{\mathcal{I}} \mid r[U] = o\} \leq \text{CMAX}(C, R, U)$.
- *For each multiplicity constraint $\text{CARD}_A(C, A)$, then:*
  $o \in C^{\mathcal{I}} \rightarrow \text{CMIN}(C, A) \leq \#\{(o, a) \in A^{\mathcal{I}}\} \leq \text{CMAX}(C, A)$.
- *For all $C, C_1, \ldots, C_n \in \mathcal{C}$: if $\{C_1, \ldots, C_n\} \text{ DISJ}_C C$, then,*
  $\forall i \in \{1, \ldots, n\}.C_i \text{ ISA}_C C \land \forall j \in \{1, \ldots, n\},\ j \neq i.C^{\mathcal{I}}_i \cap C^{\mathcal{I}}_j = \emptyset$.
- *For all $R_1, \ldots, R_n \in \mathcal{R}$: if $\{R_1, \ldots, R_n\}\text{DISJ}_R$ then $\forall i \in \{1, \ldots, n\}.R^{\mathcal{I}}_i \cap R^{\mathcal{I}}_j = \emptyset$.*

- *For all* $C, C_1, \ldots, C_n \in \mathcal{C}$*: if* $\{C_1, \ldots, C_n\}$ COVER$_C$ C*, then,* $\forall i \in \{1, \ldots, n\}.C_i$ ISA$_C$ $C \wedge C^{\mathcal{I}} = \bigcup_{i=1}^{n} C_i^{\mathcal{I}}$.
- *For each* $C \in \mathcal{C}, A \in \mathcal{A}$ *such that* KEY$(C) = A$*, then* A *is an attribute and* $\forall a \in \Delta_D^{\mathcal{I}}.\#\{o \in C^{\mathcal{I}} \mid \langle o, a \rangle \in A^{\mathcal{I}}\} \leq 1$.
- *For each* $C \in \mathcal{C}$, $R_h \in \mathcal{R}$, $h \geq 1$, REL$(R_h) = \{U : C, U_1 : C_1, \ldots, U_k : C_k\}$, $k \geq 1$, $k+1$ *the arity of* R_h*, such that* EXTK$(C) = \{[U_1]R_1, \ldots, [U_h]R_h\}$*, then for all* $o_a, o_b \in C^{\mathcal{I}}$ *and for all* $t_1, s_1 \in R_1^{\mathcal{I}}$*, ...,* $t_h, s_h \in R_h^{\mathcal{I}}$ *we have that:*

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

  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* $R \in \mathcal{R}$, $U_i, U_j \in \mathcal{U}$*, for* $i \geq 2$, $i \neq j$, REL$(R) = \{U_1 : C_1, \ldots, U_i : C_i, U_j : C_j\}$, FD$(R) = \langle U_1, \ldots, U_i \rightarrow U_j \rangle$*, then for all* $t, s \in R^{\mathcal{I}}$*, we have that* $t[U_1] = s[U_1], ..., t[U_i] = s[U_i]$ *implies* $t_j = s_j$.
- *For each* $R, r_1, \ldots, r_n \in \mathcal{R}$, $R', C_1, \ldots, C_n \in \mathcal{C}$, $U_1, \ldots, U_n, u_{s1}, \ldots, u_{sn}, u_{t1}, \ldots, u_{tn} \in \mathcal{U}$, REL$(R) = \{U_1 : C_1, \ldots, U_n : C_n\}$, OBJ$(R) = R'$, ROBJ$(U_i) = r_i$, REL$(r_i) = \{u_{s1} : R', u_{t1} : C_i\}$, $2 \leq i \leq n$, EXTK$(R') = \{u_{s1}[r_1], \ldots, u_{sn}[r_n]\}$, CARD$(R', r_i, u_{si}) = (1, 1)$, CARD$(C_i, r_i, u_{ti}) = (0, 1)$, REL, CARD*, and* EXTK *interpreted as above, then* $\forall i \in \{2, \ldots, n\}.\{U_i, u_{si}, u_{ti} \in U^{\mathcal{I}} \wedge r, r_i \in R^{\mathcal{I}} \wedge o_i, r' \in C^{\mathcal{I}} \mid u_{si} \in U^{\mathcal{I}} \rightarrow$ PLAYER$(R, U) = r' \wedge u_{ti} \in U^{\mathcal{I}} \rightarrow$ PLAYER$(R, U) = o_i\}$.
- *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* REL$(R_i) = \{U_i : C_i, \ldots U_m : C_m\}$ *(and, thus,* $R_i \in R_i^{\mathcal{I}}$ *and* $o_j \in C_j^{\mathcal{I}}$*), if* $\{U_1, U_2, \ldots U_{i-1}\}$ REX U_i*, then* $\forall i \in \{1, \ldots, i\}.o_j \in C_j^{\mathcal{I}} \rightarrow$ CMIN$(o_j, r_i, u_i) \leq 1 \wedge u_i \neq u_1 \wedge \ldots \wedge u_i \neq u_{i-1}$ *where* $u_i \in U_i^{\mathcal{I}}$, $r_i \in R_i^{\mathcal{I}}$.
- *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* REL$(R_i) = \{U_i : C_i, \ldots U_m : C_m\}$*, if* $\{U_1, U_2, \ldots U_{i-1}\}$ RDM U_i*, then* $\forall i \in \{1, \ldots, n\}.o_j \in C_j^{\mathcal{I}} \rightarrow$ CMIN$(o_j, r_i, u_i) \geq 1$ *where* $u_i \in U_i^{\mathcal{I}}$, $r_i \in R_i^{\mathcal{I}}$.

We summarize how $\mathcal{DLR}_{ifd}$ can capture conceptual models expressed in $\mathcal{CM}_{com}$, following the same approach as [1, 36] and extended with the new features.

Definition 3 (Mapping $\mathcal{CM}_{com}$ into $\mathcal{DLR}_{ifd}$) *Let* $\Sigma = (\mathcal{L}$, REL, ATT, CARD$_R$, CARD$_A$, ISA$_C$, ISA$_R$, ISA$_U$, DISJ$_C$, COVER$_C$, DISJ$_R$, KEY, EXTK, FD, OBJ, REX, RDM$)$ *be a* $\mathcal{CM}_{com}$ *conceptual data model. The* $\mathcal{DLR}_{ifd}$ *knowledge base,* $\mathcal{K}$*, mapping* Σ *is as follows.*

- *For each* $A \in \mathcal{A}$*, then,* $A \sqsubseteq \texttt{From}{:}\top \sqcap \texttt{To}{:}\top \in \mathcal{K}$*;*
- *If* C_1 ISA$_C$ $C_2 \in \Sigma$*, then,* $C_1 \sqsubseteq C_2 \in \mathcal{K}$*;*
- *If* R_1 ISA$_R$ $R_2 \in \Sigma$*, then,* $R_1 \sqsubseteq R_2 \in \mathcal{K}$*;*
- *If* U_1 ISA$_U$ $U_2 \in \Sigma$*, then* $\mathcal{K}$ *contains:* $[U_1]R_1 \sqsubseteq [U_2]R_2$*;* $R_1 \sqsubseteq \neg R_2$*;*
- *If* REL$(R) = \{U_1 {:} C_1, \ldots, U_k {:} C_k\} \in \Sigma$*, then* $R \sqsubseteq U_1 {:} C_1 \sqcap \ldots \sqcap U_k {:} C_k \in \mathcal{K}$*;*

- *If* $\text{ATT}(C) = \{A_1 : D_1, \ldots, A_h : D_h\} \in \Sigma$, *then,* $C \sqsubseteq \exists[\texttt{From}]A_1 \sqcap \ldots \sqcap \exists[\texttt{From}]A_h \sqcap \forall[\texttt{From}](A_1 \rightarrow \texttt{To} : D_1) \sqcap \ldots \sqcap \forall[\texttt{From}](A_h \rightarrow \texttt{To} : D_h) \in \mathcal{K}$;
- *If* $\text{CARD}_C(C, R, U) = (m, n) \in \Sigma$, *then,* $C \sqsubseteq \exists^{\geq m}[U]R \sqcap \exists^{\leq n}[U]R \in \mathcal{K}$;
- *If* $\text{CARD}_A(C, A) = (m, n) \in \Sigma$, *then,* $C \sqsubseteq \exists^{\geq m}[U]R \sqcap \exists^{\leq n}[U]R \in \mathcal{K}$;
- *If* $\{C_1, \ldots, C_n\}$ DISJ_C $C \in \Sigma$, *then* $\mathcal{K}$ *contains:* $C_1 \sqsubseteq C \sqcap \neg C_2 \sqcap \ldots \sqcap \neg C_n$, $C_2 \sqsubseteq C \sqcap \neg C_3 \sqcap \ldots \sqcap \neg C_n$, ..., $C_n \sqsubseteq C$;
- *If* $\{R_1, ..., R_n\}\text{DISJ}_R \in \Sigma$, *then* $\mathcal{K}$ *contains:* $R_1 \sqsubseteq \neg R_2 \sqcap \ldots \sqcap \neg R_n$, $R_2 \sqsubseteq \neg R_3 \sqcap \ldots \sqcap \neg R_n$, ..., $R_{n-1} \sqsubseteq \neg R_n$;
- *If* $\{C_1, \ldots, C_n\}$ COVER_C $C \in \Sigma$, *then* $\mathcal{K}$ *contains:* $C_1 \sqsubseteq C$, ..., $C_n \sqsubseteq C$; $C \sqsubseteq C_1 \sqcup \ldots \sqcup C_n$;
- *If* $\text{KEY}(C) = A \in \Sigma$, *then,* $\mathcal{K}$ *contains:* $C \sqsubseteq \exists^{=1}[\texttt{From}]A$; $\top \sqsubseteq \exists^{\leq 1}[\texttt{To}](A \sqcap [\texttt{From}] : C)$;
- *If* $\text{EXTK}(C) = \{[U_1]R, \ldots, [U_h]R_h\} \in \Sigma$, *then* $\mathcal{K}$ *contains:* $(\mathbf{id}\ C\ [U_1]R_1, \ldots, [U_h]R_h)$;
- *If* $\text{FD}(R) = \langle U_1, \ldots, U_i \rightarrow j\rangle \in \Sigma$, *then* $\mathcal{K}$ *contains:* $(\mathbf{fd}\ R\ U_1, \ldots, U_i \rightarrow j)$;
- *If* $\text{OBJ}(R) = R'$, *then* $\mathcal{K}$ *contains:* $(\mathbf{id}\ R'\ [u_{s1}]r_1, \ldots, [u_{sn}]r_n)$;
$$\begin{aligned} R' \sqsubseteq\ & \exists[u_{s1}]r_1 \sqcap (\leq 1[u_{s1}]r_1) \sqcap \forall[u_{s1}](r_1 \Rightarrow (u_{t1} : C_1))\sqcap \\ & \exists[u_{s2}]r_2 \sqcap (\leq 1[u_{s2}]r_2) \sqcap \forall[u_{s2}](r_2 \Rightarrow (u_{t2} : C_2))\sqcap \\ & \ldots \\ & \exists[u_{sn}]r_n \sqcap (\leq 1[u_{sn}]r_n) \sqcap \forall[u_{sn}](r_n \Rightarrow (u_{tn} : C_n)); \end{aligned}$$
- *If* $\{U_1, U_2, \ldots U_{i-1}\}$ REX $U_i \in \Sigma$, *then* $\mathcal{K}$ *contains:* $C \sqsubseteq (\exists^{\leq 1}[U_1]R_1 \sqcup \ldots \sqcup \exists^{\leq 1}[U_i]R_i)$; $[U_1]R_1 \sqsubseteq \neg[U_2]R_2 \sqcap \ldots \sqcap \neg[U_{i-1}]R_{i-1}$, $[U_2]R_2 \sqsubseteq \neg[U_3]R_3 \sqcap \ldots \sqcap \neg[U_{i-1}]R_{i-1}$, ..., $[U_{i-1}]R_{i-1} \sqsubseteq \neg[U_i]R_i$;
- *If* $\{U_1, U_2, \ldots U_{i-1}\}$ RDM $U_i \in \Sigma$, *then* $\mathcal{K}$ *contains:* $C \sqsubseteq (\exists^{\geq 1}[U_1]R_1 \sqcup \ldots \sqcup \exists^{\geq 1}[U_i]R_i)$; $[U_1]R_1 \sqsubseteq \neg[U_2]R_2 \sqcap \ldots \sqcap \neg[U_{i-1}]R_{i-1}$, $[U_2]R_2 \sqsubseteq \neg[U_3]R_3 \sqcap \ldots \sqcap \neg[U_{i-1}]R_{i-1}$, ..., $[U_{i-1}]R_{i-1} \sqsubseteq \neg[U_i]R_i$;

Thus, $\mathcal{CM}_{com}$ has an equi-satisfiable $\mathcal{DLR}_{ifd}$ knowledge base and therewith we can avail of the nice computational properties of $\mathcal{DLR}_{ifd}$ [9, 13]. One could have chosen another expressive DL language as formal foundation, such as OWL 2 DL [50] with a corresponding mapping to a $\mathcal{CM}'_{com}$. However, such a $\mathcal{CM}'_{com}$ would have REL restricted to binaries, it would not have EXTK, FD, OBJ, REX, and RDM, but gained the option to represent transitivity, reflexivity, irreflexivity, asymmetry, and symmetry. Those gains with the relational properties, however, are useful only to formalize ORM's ring constraints and come at the cost of 2NExpTime complexity in concept and theory satisfiability ($\mathcal{DLR}_{ifd}$ is in ExpTime). There are always tradeoffs in a formalization, and the priority here is being able to deal with the core features of conceptual data modeling languages before extending one's horizon. As we shall see in Section 1.2.2, $\mathcal{CM}_{com}$ is the greatest common denominator by capturing most or all features of EER, UML Class Diagrams, and ORM.

1.2.2 EER, UML, and ORM in terms of $\mathcal{CM}_{com}$

With the formal apparatus in place, we now can consider definitions of EER, UML, and ORM 2 in terms of $\mathcal{CM}_{com}$. The rationale for the exact combination of constraints has been explained and discussed in detail elsewhere [36] (e.g., regarding UML's OCL [55], aggregation [41], and identification [40]). The "$^-$" in "$_{ORM2^-}$" is due to, mainly, ORM's undecidability due to constraints over k roles over an

n-ary relation, $n \geq 3$, and $k < n$ [38], and the unknown computational complexity of antisymmetry (the fine-grained arguments are beyond the current scope). The important point here is to have basic definitions so as to focus on language features and the ontology-driven aspects.

Definition 4 **($\mathcal{CM}_{EER}$)** *A $\mathcal{CM}_{EER}$ conceptual data model is a tuple*
$\Sigma = (\mathcal{L}, \text{REL}, \text{ATT}, \text{CARD}_R, \text{ISA}_C, \text{DISJ}_C, \text{COVER}_C, \text{KEY}, \text{EXTK})$
adhering to $\mathcal{CM}_{com}$ syntax and semantics.

Definition 5 **($\mathcal{CM}_{UML}$)** *A $\mathcal{CM}_{UML}$ conceptual data model is a tuple*
$\Sigma = (\mathcal{L}, \text{REL}, \text{ATT}, \text{CARD}_R, \text{ISA}_C, \text{ISA}_R, \text{DISJ}_C, \text{COVER}_C, \text{EXTK}, \text{FD}, \text{OBJ}, \text{PW})$
adhering to $\mathcal{CM}_{com}$ syntax and semantics, except for the aggregation association PW*, with syntax* $\text{PW} = \{U_1 : C_1, U_2 : C_2\}$*, that has no defined semantics.*

Definition 6 *A $\mathcal{CM}_{ORM2^-}$ conceptual data model is a tuple*
$\Sigma = (\mathcal{L}, \text{REL}, \text{ATT}, \text{CARD}_R, \text{CARD}_A, \text{ISA}_C, \text{ISA}_R, \text{ISA}_U, \text{DISJ}_C, \text{COVER}_C, \text{KEY}, \text{EXTK},$
$\text{FD}, \text{OBJ}, \text{DISJ}_R, \text{REX}, \text{RDM})$
adhering to $\mathcal{CM}_{com}$ syntax and semantics.

There is a notable difference between $\mathcal{CM}_{EER}$, $\mathcal{CM}_{UML}$, and $\mathcal{CM}_{ORM2^-}$. Although it is possible to include ISA_U for UML's association ends in $\mathcal{CM}_{UML}$ and interpret the OMG standard [52] liberally on keys (KEY and EXTK; see [40] for a discussion) and attributes (by modeling them external to the class so that one can use CARD_A), this is yet to be refined in the standard and implemented in the CASE tools. From an HCI perspective, a less expressive language may be pleasing for a novice modeler, but it is worth noting that extensions have been deemed necessary [34] and were proposed for EER to address better the requirements of molecular biologists [21] (discussed in Section 1.3). Here we illustrate two modeling aspects—role exclusion and ternaries—and their solutions with $\mathcal{CM}_{com}$.

Example 2 (Modeling features in the graphical and formal languages) Let us take Kazic's complaint on not being able to represent thymidine phosphorylase binding with thymidine or phosphate [33]. This requires an exclusion constraint over roles (REX), which is possible to represent with ORM and $\mathcal{CM}_{ORM2^-}$, as demonstrated in Figure 1.4-A, but not in the EER or UML graphical syntax. Formally, we have $\{\texttt{bindsT}, \texttt{bindsP}\}$ REX `binds` in $\mathcal{CM}_{com}$, and thus the $\mathcal{DLR}_{ifd}$ knowledge base $\mathcal{K}$ contains
$\texttt{ThymidinePhosphorylase} \sqsubseteq (\exists^{\leq 1}[\texttt{bindsT}]\texttt{binds}_1 \sqcup \exists^{\leq 1}[\texttt{bindsP}]\texttt{binds}_2)$,
$[\texttt{bindsT}]\texttt{binds}_1 \sqsubseteq \neg[\texttt{bindsP}]\texttt{binds}_2$.
Clearly, the formal foundation now underpinning EER and UML with $\mathcal{CM}_{EER}$ and $\mathcal{CM}_{UML}$ permits one to add such constraints to the respective graphical language.

The widely noted OWL shortcoming of n-ary relationships with $n \geq 3$ can be handled easily in $\mathcal{CM}_{com}$, because we can with $\mathcal{DLR}_{ifd}$. Figure 1.4-B depicts a ternary for recording epidemiological data on the path of infection of particular HIV subtypes from Donor to Recipient, which is represented in $\mathcal{CM}_{com}$ as $\text{REL}(\texttt{HIVtransmission}) = \{\texttt{object} : \texttt{HIVsubtype}, \texttt{from} : \texttt{Donor}, \texttt{to} : \texttt{Recipient}\}$, where `object`, `from` and `to` are the roles played by the objects (not depicted in Figure 1.4-B). The corresponding translations into $\mathcal{DLR}_{ifd}$ is:
$\texttt{HIVtransmission} \sqsubseteq [\texttt{object}]\texttt{HIVsubtype} \sqcap [\texttt{from}]\texttt{Donor} \sqcap [\texttt{to}]\texttt{Recipient}$. ◇

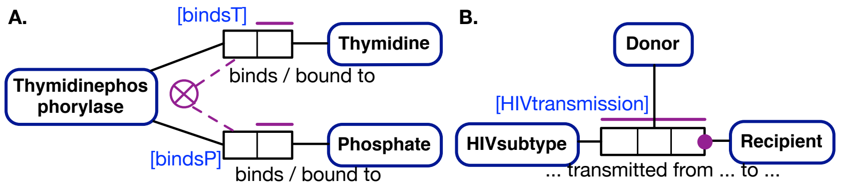


Figure 1.4 ORM2 examples. A: role exclusion, which cannot be represented in standard UML and EER; B: ternary relation, which cannot be represented fully in OWL.

Such differences in both graphical syntax and the underlying formalization are being investigated [1, 9, 16, 36, 38, 55] and proof-of-concept implementations exist [22, 56]. What they have in common is the use of a *decidable* logic language so as to guarantee that the reasoning services terminate. We will look at several reasoning scenarios in Section 1.4.1.

1.3 EXTENSIONS

The need for extensions of conceptual data modeling languages clearly depends on which language one chooses to extend, because each one differs in expressiveness, as we have seen in the previous section. In addition to the expressiveness within $\mathcal{DLR}_{ifd}$—say, adding KEY or ISA$_U$ to $\mathcal{CM}_{UML}$ (hence, to UML)—one can add features to the language that go beyond $\mathcal{DLR}_{ifd}$ or even beyond first order predicate logic. Such extensions are motivated by the identification of *what* to represent and *how* in order to better capture the subject domain semantics. We shall look first at the former, which concerns incorporating notions of Ontology that generally require extensions to the language, which will be addressed afterward.

1.3.1 Ontology-driven modeling

The case for ontology-driven conceptual data modeling is described in some detail in [26, 27]: notions from philosophy can be used to solve modeling issues, improve conceptual data models, and provide explanations why one representation of a piece of information is better than another. Methodologically, this can be done by (*i*) providing a solution to a recurring modeling problem, (*ii*) using an ontology to generate several conceptual data models (preliminary solutions are described in [20, 32, 61]), and (*iii*) integrating (a section of) an ontology into the conceptual data model that subsequently is converted into data in the database; they were depicted in bold-face with respect to the traditional waterfall methodology of database and software development in Figure 1.2. The first two options are young fields of research, whereas the third option is used widely in bioinformatics databases where, e.g., the Kyoto Encyclopedia of Genes and Genomes (KEGG) [44] or the Gene Ontology (GO) [24] are used for annotation of gene products, thereby linking primary and boutique databases, such as Uniprot [62] and the Horizontal Gene Transfer DataBase (HGT-DB) [23].

Option (*i*) considers (re-)usable components of foundational ontologies such as the Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE) [49], Basic Formal Ontology (BFO) [10], or General Formal Ontology (GFO) [31], being the high-level categories and generic relationships, such as the distinction between endurants and perdurants, and relationships like parthood, participation, and dependency. It provides modeling guidance, and informs and refines language features, such as relational properties like transitivity, positionalism of relations, and identification mechanisms. In all cases, it offers answers how to best represent some piece of information and provides justifications why. The most prominent results of ontology-driven modeling to solve a recurring modeling issue is that of part-whole relations [3, 4, 29, 35, 41]. These refinements deal both with clarifying the different types of part-whole relations and how the entities participate the relation. The part-whole relations can be structured in a hierarchy [41], which, in turn, contributes to correct usage and deductions in the conceptual data model and, if linked to data, enhances capabilities for information retrieval.

Example 3 (Ontology-driven conceptual data modeling: parthood) A cell nucleus is spatially *contained in* an eukaryotic cell, but not a structural part of it, whereas the region occupied by the nucleus is proper part of the region occupied by the eukaryotic cell it contains (proper parthood implies proper containment, but not vice versa). A cell receptor is a *structural part of* a cell wall's lipid bi-layer but not a proper part, because part of the receptor is external to it.

Let $\textsc{rel}(\texttt{hasStructuralPart}) = \{\texttt{whole} : \texttt{3-Chlorobenzoate}, \texttt{part} : \texttt{Benzene}\}$ and $\texttt{hasStructuralPart}\ \textsc{isa}_{\texttt{R}}\ \texttt{hasPart}$, then a query "retrieve all molecules that `hasPart` some `Benzene`" will have in the query answer `3-Chlorobenzoate` despite it not being represented explicitly: it is inferred thanks to the $\textsc{isa}_{\texttt{R}}$ assertion. A query "retrieve all molecules that `hasStructuralPart` some `Benzene`" will *not* return enzyme-substrate complexes involving benzene rings where the benzene ring is spatially contained in the 'hole' or lock of the receptor in the receptor-ligand interaction; hence, with a differentiation between parthood and containment (among others), the query answer will not contain false positives or noisy information. ♢

In addition to refining particular relationships, there are guidelines from Ontology and AI to design a good taxonomy, such as using ideas from the OntoClean method [28]. OntoClean relies on meta-level properties, such as rigidity, and the foundational ontologies help identifying the nature of the class or relationship under consideration. For instance, Protein has the property of being rigid (each individual object that is member of a Protein class is a protein for its entire existence), whereas Enzyme is anti-rigid (each individual enzyme is not necessarily always an enzyme), and an anti-rigid class cannot subsume a rigid one. Hence, one should *not* have $\texttt{Protein}\ \textsc{isa}_{\texttt{C}}\ \texttt{Enzyme}$ in the model, but $\texttt{Enzyme}\ \textsc{isa}_{\texttt{C}}\ \texttt{Protein}$ instead. Digging deeper, we see that the essential property of an enzyme is to catalyze a reaction, which is the *function* or *role* that the molecule performs, and it may be that at some point in time that protein still exists but is somehow defective in its functioning as enzyme. Similarly, one might encounter $\texttt{Tetanospasmin}\ \textsc{isa}_{\texttt{C}}\ \texttt{Zinc-Endopeptidase}$ and $\texttt{Zinc-Endopeptidase}\ \textsc{isa}_{\texttt{C}}\ \texttt{Toxin}$ in a conceptual data model: tetanospasmin is indeed a zinc-endopeptidase and a toxin produced by *Clostridium tetani*, but it only has the role of being a toxin in humans, as *C. tetani* uses the enzyme in its natural functioning of the cell. A solution pattern to better model this type of information is provided by a foundational ontology: one creates a hierarchy for rigid properties and one for anti-rigid ones that, in turn, *inhere in* or *depend on* the rigid ones, thereby distinguishing between what it structurally *is* and what it functionally *does*, which leads to an assertion like $\textsc{rel}(\texttt{inheresIn}) = \{\texttt{role} : \texttt{Enzyme}, \texttt{bearer} : \texttt{Protein}\}$. Fortunately, one does not have to start from scratch anymore with such analyses in the subject domain of molecular biology, as several ontologies exist that take this approach (e.g., BioTop [8]), which can be used in the way as outlined in option (*ii*) in the introduction of this section. Overall, this will result in better conceptual data models, which is illustrated in the next example for catalytic reactions.

Example 4 (Ontology-driven conceptual data modeling: catalysis) Catalysis has three principal participants: molecule(s) in, out, and the enzyme that catalyzes the reaction. Elmasri *et al.* [21] proposes a "process relationship" in EER to represent the static (atemporal) aspects of chemical processes, which is depicted in Figure 1.5-A, which can be scaled up trivially to multiple inputs, outputs, and/or catalysts. However, the three entities are all molecules and, more importantly, some molecule can be both an `Einput` and an `Eoutput` in different reactions or the `Eoutput` molecule is also the `Ecatalyst` (autocatalytic reaction). Modeling the molecule's roles in a certain situation as different entity types results in duplication of data, which, in turn, leads to inconsistencies or 'dirty' data in the database or application. The minimalist approach that avoids these problems is shown in Figure 1.5-B, but it is unsatisfactory due to its lack of detail. To really solve it, we use a foundational ontology for modeling guidance, and the Basic Formal Ontology [10] and Relation Ontology [60] in particular. Then, the molecule's structural characteristics—like being a Protein consisting of amino acids—are distinguished from the role it plays—like Enzyme—and matched with entity types in BFO: $\texttt{Protein}\ \textsc{isa}_\texttt{C}\ \texttt{Molecule}$, $\texttt{Molecule}\ \textsc{isa}_\texttt{C}\ \texttt{Object}$, and $\texttt{Enzyme}\ \textsc{isa}_\texttt{C}\ \texttt{Role}$, which are related through a refinement of the RO `inherence` relation such that $\texttt{role}\ \textsc{isa}_\texttt{R}\ \texttt{inherence}$, $\textsc{rel}(\texttt{role}) = \{\texttt{role} : \texttt{Enzyme}, \texttt{bearer} : \texttt{Protein}\}$, and $\textsc{card}(\texttt{Enzyme}, \texttt{role}, \texttt{role}) = 1..1$. `Object`'s superclass in BFO is `IndependentContinuant` (`IC`), and `Role`'s superclasses are `RealizableEntity`, `SpecificallyDependentContinuant`, and `DependentContinuant`, and, more generally with respect to the RO, $\textsc{rel}(\texttt{inherence}) = \{\texttt{role} : \texttt{DC}, \texttt{bearer} : \texttt{IC}\}$. Practically in the conceptual data model, the two branches from the BFO hierarchy can be added in whole or in abbreviated form, which is indicated with the dashed lines and rectangles in Figure 1.5-C. This conceptual model fragment already solves the problem of data duplication that would occur with a database based on the conceptual model from Figure 1.5-A and facilitates querying such that the query answer will have less noisy results (see also Examples 3 and 6). ♢

1.3.2 More expressive languages

Multiple extensions to conceptual data modeling languages exist, of which the most interesting one from a molecular biology perspective is the temporal dimension, i.e., to have a means to specify unambiguously what changes, how, and under which conditions. This can be an object instantiating different classes at different points

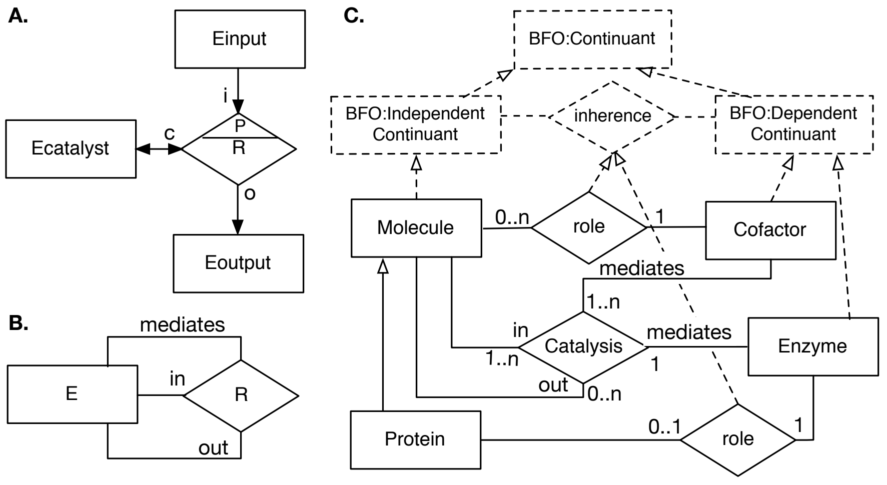


Figure 1.5 Static aspects of modeling single processes (catalytic reactions) in EER. A: Elmasri *et al.*'s [21] proposal, with input, output and catalyst molecules; B: The essential roles played; C: An example of a more refined representation of catalysis, informed by ontology, where the dashed entities and subsumption relationships are a fragment of BFO.

in time (called *object migration*), attributes that hold for a specific duration, or the relation between objects changes (*relation migration*). For instance, each immune system cell Macrophage must have been an instance of Monocyte beforehand, i.e., it "dynamically evolved" (DEV$^-$) [37], the value for DNA's attribute hasFoldingState changes to unfolded only for the duration that it is being transcribed, and the enzyme inheres in the molecule that is also a substrate in an autocatalytic reaction, so the interaction "dynamically extends" (RDEX) [42] to autoCatalysis, where RDEX can be defined using a temporal interpretation [42]:

$$\mathtt{R}\ \textsc{Rdex}\ \mathtt{R}' \text{ if and only if } \langle o_1, o_2\rangle \in \mathtt{R}^{\mathcal{I}(t)} \to \exists t' > t.\langle o_1, o_2\rangle \in \mathtt{R}'^{\mathcal{I}(t')} \qquad (1.1)$$

where $\mathtt{R}, \mathtt{R}'$ are relationships, $t, t' \in \mathcal{T}_p$ and $\mathcal{T}_p$ is a set of time points, and $\cdot^{\mathcal{I}(t)}$ the interpretation function for a given snapshot of the state of affairs at that time. An elegant extension to $\mathcal{CM}_{com}$ can handle this (introduced below), so that the conceptual data model contains assertions such as $\textsc{rel}(\mathtt{role}) = \{\mathtt{bearer} : \mathtt{RNAmolecule}, \mathtt{role} : \mathtt{Ribozyme}\}$ and $\textsc{rel}(\mathtt{autoCatalysis}) = \{\mathtt{substrate} : \mathtt{RNAmolecule}, \mathtt{catalyst} : \mathtt{Ribozyme}\}$, and such that `role` RDEX `autoCatalysis` holds. Another typical challenge is how to represent metabolic pathways or genetics' Central Dogma with its part-processes in a specific sequence.

To cater for the representation of this kind of information, the first questions to answer are: which fundamental aspects involving time have to be represented, which language features does it require, and how to do that in a conceptual data model in such a way that it can be stored by the software and checked on consistency? Extant proposals include adding an ordered bag to EER to represent a notion of sequence of events [21] and using UML's sequence and activity diagrams (to model SARS-CoV infection) [58] that can represent processual information, although the

UML diagrams have no formal foundation. Both proposals, however, do not let one represent concurrent reactions, detect inconsistencies (e.g., contain a cycle where there should not be one), or derive implicit information about such dynamic information. To be able to do so, one has to be able to represent the reactions as n-ary relationships, which can be represented in $\mathcal{CM}_{com}$, and formalize notions such as 'precedes', 'during' and similar natural language terms, which requires a language extension. For instance, we have to include somehow that "a immediately precedes b" means that we have not only (a, t) and (b, t') but also for each time point that $\neg\exists t''.t < t'' < t'$ and $t \neq t'$, where $t, t', t'' \in \mathcal{T}_p$. One can formalize "$\mathsf{a}$ during b" and the other 11 Allen temporal relations using the same approach. MADS [53] for spatio-temporal conceptual data modeling is fairly comprehensive, though its inclusion of temporal knowledge representation has been extended informed by a $\mathcal{DLR_{US}}$ foundation that also has a mapping to the temporally extended EER, called $\mathcal{ER}_{VT}$ [5]. $\mathcal{DLR_{US}}$ is in the same DL family as $\mathcal{DLR}_{\mathit{ifd}}$, but it does not have the "ifd" features—hence, compared to $\mathcal{CM}_{com}$, it will not have EXTK, FD, and OBJ—and instead has temporal classes, relationships, attributes, and evolution constraints to specify what changes, and how. Let us briefly illustrate this for temporal classes. A fragment of the $\mathcal{DLR_{US}}$ syntax is:

$$\begin{array}{rcl} C & \rightarrow & \top \mid \bot \mid A \mid \neg C \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2 \mid \exists^{\leqslant k}[U_j]R \mid \\ & & \diamond^+ C \mid \diamond^- C \mid \Box^+ C \mid \Box^- C \mid \oplus C \mid \ominus C \mid C_1 \,\mathcal{U}\, C_2 \mid C_1 \,\mathcal{S}\, C_2 \end{array}$$

where the first line is the same as what we have seen for $\mathcal{DLR}$ and the second line introduces the temporal operators. The semantics of these operators are as follows. First, the $\mathcal{U}$ntil and $\mathcal{S}$ince operators, where $(u, v) = \{w \in \mathcal{T} \mid u < w < v\}$:
$(C_1 \,\mathcal{U}\, C_2)^{\mathcal{I}(t)} = \{\, d \in \top^{\mathcal{I}(t)} \mid \exists v > t.(d \in C_2^{\mathcal{I}(v)} \wedge \forall w \in (t, v).d \in C_1^{\mathcal{I}(w)})\}$;
$(C_1 \,\mathcal{S}\, C_2)^{\mathcal{I}(t)} = \{\, d \in \top^{\mathcal{I}(t)} \mid \exists v < t.(d \in C_2^{\mathcal{I}(v)} \wedge \forall w \in (v, t).d \in C_1^{\mathcal{I}(w)})\}$.
Second, $\mathcal{U}$ and $\mathcal{S}$ together with $\bot$ and $\top$ suffice to define the other ones: the temporal operator $\diamond^+$ (some time in the future) as $\diamond^+ C \equiv \top \,\mathcal{U}\, C$, $\oplus$ (at the next moment) as $\oplus C \equiv \bot \,\mathcal{U}\, C$, and likewise for their past counterparts $\diamond^-$ (some time in the past) as $\diamond^- C \equiv \top \,\mathcal{S}\, C$ and $\ominus$ (at the previous moment) as $\ominus C \equiv \bot \,\mathcal{S}\, C$. The operators $\Box^+$ (always in the future) and $\Box^-$ (always in the past) are the duals of $\diamond^+$ and $\diamond^-$ (some time in the past), respectively, i.e., $\Box^+ C \equiv \neg\diamond^+\neg C$ and $\Box^- C \equiv \neg\diamond^-\neg C$, and, finally, the operators $\diamond^*$ (at some moment) and its dual $\Box^*$ (at all moments) can be defined as $\diamond^* C \equiv C \sqcup \diamond^+ C \sqcup \diamond^- C$ and $\Box^* C \equiv C \sqcap \Box^+ C \sqcap \Box^- C$, respectively. This is similar for relationships in $\mathcal{DLR_{US}}$. Then, we can perform the same procedure as for $\mathcal{CM}_{com}$: generate a fixed textual version for $\mathcal{CM}_{com}^-$+temporal extension, ${}^t\mathcal{CM}$ (alike Definition 1), declare a mapping from ${}^t\mathcal{CM}$ to a graphical syntax for the temporal operators, fix the semantics (alike Definition 2), and declare a mapping (alike Definition 3) to show that for each ${}^t\mathcal{CM}$ there is an equi-satisfiable $\mathcal{DLR_{US}}$ knowledge base. This has been done already for EER without EXTK, FD and OBJ [5] (named $\mathcal{ER}_{VT}$), which we shall not repeat here, but illustrate with the aforementioned examples. The "RDEX" of aforementioned assertion `role` RDEX `autoCatalysis` can be added to the language's syntax (to a Definition 1′ for ${}^t\mathcal{CM}$), the semantics as in Eq. 1.1 aded to a Definition 2′, and a mapping into $\mathcal{DLR_{US}}$ as $\mathtt{R} \sqsubseteq \diamond^+\mathtt{R}'$ added to a Definition 3′. The dynamic evolution constraint for classes is represented syntactically as `C` DEV `C′`, has a semantics of $o \in \mathtt{C}^{\mathcal{I}(t)} \rightarrow \exists t' > t.o \in \mathtt{C'}^{\mathcal{I}(t')} \wedge o \notin \mathtt{C}^{\mathcal{I}(t')}$, and is mapped into

$\mathcal{DLR}_{\mathcal{US}}$ as $\texttt{C} \sqsubseteq \Diamond^{+}(\texttt{C}' \sqcap \neg\texttt{C})$. We now also can distinguish between the "a precedes b", i.e., a at holds at *some time* before b, $\Diamond^{-}$, and "a immediately precedes b", $\ominus$. This extension is particularly useful in molecular biology for modeling (and, as we will see later, checking consistency of) metabolic and biosynthetic pathways, which is illustrated in the next example.

Example 5 (Language extensions: temporal) We can use the DEV evolution constraint to represent formally a meaningful relation between afore-mentioned Monocyte and Macrophage: in the normal course of things, each monocyte transforms into a macrophage, but such a cell is never both at the same time, hence, the conceptual data model has Monocyte DEV Macrophage (i.e., $\texttt{Monocyte} \sqsubseteq \Diamond^{+}(\texttt{Macrophage} \sqcap \neg\texttt{Monocyte})$ in the $\mathcal{DLR}_{\mathcal{US}}$ knowledge base); it is trivial to model this the other way around with DEV^{-} (that each macrophage must have been a monocyte earlier).

Let us now consider SARS viral infection events [58] that informally asserts that first the virus binds to the receptor, then either the membranes fuse or the virus detaches from the cell, and the virus enters the cell only after membrane fusion. Informed by a foundational ontology, one can create either a hierarchy of processes (Binds $\text{ISA}_{\texttt{C}}$ Process etc.) and such that each process has participants (REL(hasParticipant) $= \{$process : Binds, participant : Virus$\}$ and REL(hasParticipant) $= \{$process : Binds, participant : CellReceptor$\}$ etc,), or a more compact representation by creating relationships REL(binds) $= \{$binder : Virus, bindsTo : CellReceptor$\}$ and likewise for REL(membraneFusion), REL(detach), and REL(viralEntry). As it does not matter which option we choose thanks to having temporal operators on both classes and relations in $\mathcal{DLR}_{\mathcal{US}}$, let us take the second option. Then, given the subject domain semantics, $\texttt{ViralEntry} \sqsubseteq \Diamond^{-}\ \texttt{Binds}$ must hold ("for each viral entry, it must have been bound some time before"), but not the converse ($\texttt{Binds} \sqsubseteq \Diamond^{+}\ \texttt{ViralEntry}$) because the virus may detach. Regarding instances and tuple migration in a scenario of, say, simulations or processing annotations of video about cell processes, then the intention to add a migration from $\langle\texttt{virus1}, \texttt{cell1}\rangle \in \texttt{viralEntry}$ to $\langle\texttt{virus1}, \texttt{cell1}\rangle \in \texttt{membraneFusion}$ should result in a violation of the integrity constraint suggested by the subject domain knowledge of the database. This cannot be guaranteed in software based on plain UML, EER, or ORM, but can with the additional temporal extension; hence, it ensures that the data represents events in reality more precisely with less errors. $\Diamond$

Besides temporality, one can add, among others, fuzzy, rough, or probabilistic features (e.g., fuzzy $\mathcal{DLR}$ [47]), which offer further options for creative modeling. For instance, $\mathcal{DLR}_{\mathcal{US}}$ has been shown to be useful to define the notions of *essential* and *immutable* parts [3], the fuzzy extension can cope with inclusion of a concept like Small Molecule that has no clear cut-off point for the actual size, and probabilistic knowledge can be used to represent 'default' and 'typical' cases [45]. The tradeoff, however, is that while the additional features are great for modeling biological knowledge more precisely, it negatively affects automated reasoning services and scalability of the information systems—and it is exactly the automated reasoning services that contribute to *in silico* biological knowledge discovery.

1.4 AUTOMATED REASONING AND BIOLOGICAL KNOWLEDGE DISCOVERY

In this section, we briefly describe how various automated reasoning services and sophisticated querying can be used in the domain expert's 'toolbox' for biological knowledge discovery.

1.4.1 Exploiting automated reasoning services

A major advantage of the formal foundation for the conceptual data modeling languages with $\mathcal{DLR}_{ifd}$ is its decidability, hence, the guarantee that the reasoning services that can be deployed for *in silico* biological knowledge discovery will terminate; more precisely, $\mathcal{DLR}_{ifd}$ is ExpTime-complete, hence, so is $\mathcal{CM}_{com}$. As a basis, we can use so-called 'standard' reasoning services for checking diagram and class consistency, class subsumption reasoning, and certain other implicit consequences. We describe them here briefly in terms of DL:

- *Conceptual data model consistency*: the whole conceptual data model Σ is consistent if it admits an instantiation, i.e., all classes in $\mathcal{C}$ can be populated without violating any of the constraints; formally in $\mathcal{DLR}_{ifd}$: $\Sigma \not\models \top \sqsubseteq \bot$;
- *Class consistency*: a class $C \in \mathcal{C}$ is consistent if Σ admits an instantiation in which the class has a nonempty set of instances; formally: $\Sigma \not\models C \sqsubseteq \bot$;
- *Class subsumption*: a class C_1 subsumes a class C_2 (i.e, C_2 ISA C_1 in Σ), if Σ implies that C_1 is a generalization of C_2 (or: all instances of C_2 are also instances of C_1), with $C_1, C_2 \in \mathcal{C}$; formally: $\Sigma \models C_2 \sqsubseteq C_1$;

Berardi *et al.* [9] also include refinement of multiplicities and typing for UML Class Diagrams, which means that the interaction of properties of several related classes may results in stricter multiplicities or typing than has been specified explicitly in Σ, and this service can be applied to $\mathcal{CM}_{com}$ as well. In addition, we can avail of two more reasoning services commonly used with DL knowledge bases, but only if one somehow combines the conceptual data model Σ with the instances in the software application or database. These services are *instance classification*—is a a member of C in Σ? i.e., $\Sigma \models C(a)$—and *instance retrieval*, meaning to compute all individuals a such that $C(a)$ is satisfied by every interpretation of Σ (i.e., $\{a \mid \Sigma \models C(a)\}$).

A notable achievement in biological data discovery using automated reasoning has been the classification of protein phosphatases [63], where a novel protein phosphatase was discovered. It has also been used for finding suitable molecules in rubber manufacturing that matched the criteria (i.e., chosen attributes) [7], whose approach can be employed also in pharma-informatics when looking for drug candidates. For instance, to search for potential antibiotics by using criteria for the desired molecule such as 'has as part a β-lactam ring', 'is water-soluble', and that also 'function as enzyme inhibitor'. Automated reasoning is illustrated in the next example with enzymes, proteins, substrate and co-factors.

Example 6 (Reasoning over an ontology-driven conceptual data model) Let us take a ontology-enhanced sample UML diagram about enzymes, as shown in Figure 1.6, that is both formalized in $\mathcal{CM}_{com}$ (with $\textsc{rel}(\texttt{inheresIn}) = \{\texttt{role} : \texttt{Enzyme}, \texttt{bearer} : \texttt{Protein}\}$, $\textsc{card}(\texttt{Enzyme}, \texttt{Protein}, \texttt{inheresIn}) = (\texttt{1}, \texttt{n})$ etc.) and the enzymes, proteins, substrates, and cofactors are extracted from one or more

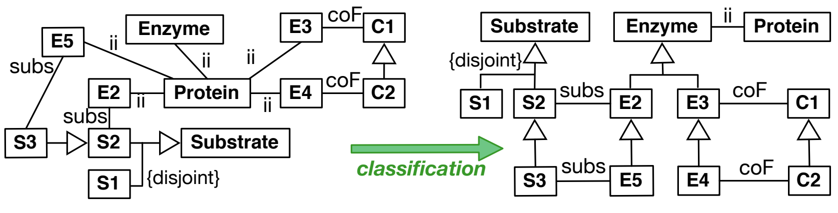


Figure 1.6 Sample $\mathcal{CM}_{UML}$ before (left) and after (right) classification (multiplicity not drawn); associations: ii: inheres in; subs: has substrate; coF: has co-factor.

ontologies—i.e., they are ontology modules adapted for conceptual data modeling; see Section 1.3.1 and Figure 1.1)—such as the PRO protein ontology [51] and Biopax [19]. Class subsumption reasoning re-orders the classes in the taxonomy according to the properties they have; e.g., for E2, we have not only $\text{REL}(\texttt{inheresIn}) = \{\texttt{role} : \texttt{E2}, \texttt{bearer} : \texttt{Protein}\}$, but also $\text{REL}(\texttt{hasSubstrate}) = \{\texttt{actor} : \texttt{E2}, \texttt{actee} : \texttt{S2}\}$, i.e., one more property than Enzyme, hence, all instances of E2 must also be instances of Enzyme in all possible models and therefore E2 ISA Enzyme holds. E5 does not have more properties, but S3 ISA S2 and therefore E5 ISA E2. Second, if the diagram would have had $\text{CARD}(\texttt{E3}, \texttt{C1}, \texttt{coF}) = (1, 1)$ and $\text{CARD}(\texttt{E4}, \texttt{C2}, \texttt{coF}) = (1, \texttt{n})$, then it would have deduced $\text{CARD}(\texttt{E4}, \texttt{C2}, \texttt{coF}) = (1, 1)$ as refined cardinality to comply with the inherited constraint. Now, if we add the hasSubstrate association to E3, then several things can happen, depending on the substrate: (*i*) if $\text{REL}(\texttt{hasSubstrate}) = \{\texttt{actor} : \texttt{E3}, \texttt{actee} : \texttt{S2}\}$, then the reasoner will deduce E3 ISA E2, (*ii*) if $\text{REL}(\texttt{hasSub- strate}) = \{\texttt{actor} : \texttt{E3}, \texttt{actee} : \texttt{S3}\}$, then E3 ISA E5, and (*iii*) if $\text{REL}(\texttt{hasSubstrate}) = \{\texttt{actor} : \texttt{E3}, \texttt{actee} : \texttt{S1}\}$, then E3 ISA Enzyme (because $\{\texttt{S1}, \texttt{S2}\}$ $\text{DISJ}_{\texttt{C}}$ Substrate).

Querying is a form of reasoning, too, which can be done over the conceptual data model itself, over a database, or their combination [14, 15]. For instance, an automated reasoner evaluates "retrieve all enzymes that have C1 as coFactor" by traversing the tree from Enzyme down to all classes that have an association coFactor with C1 as class at the other end. In the case of the conceptual data model depicted in Figure 1.6, it will return E3, E4 as answer: E3 because it is directly related and E4 because each C2 is also a C1. ◇

1.4.2 Finding new relationships and classes by using instances

Combining conceptual data models and the data in the information system is challenging and, to the best of our knowledge, no end-user usable implementations have been realized for the scenarios described in this section. The idea is that instances will justify information represented in the conceptual data model and that the model is used for analyzing instances, so that the requirements are essentially those for a knowledge base (TBox & Abox), and in particular scalability of reasoning over a conceptual data model in the presence of large amounts of data stored in a database (see, e.g., [14, 46] for preliminary results). Focusing on the advantages this option offers, there are three patterns for discovering implicit knowledge in

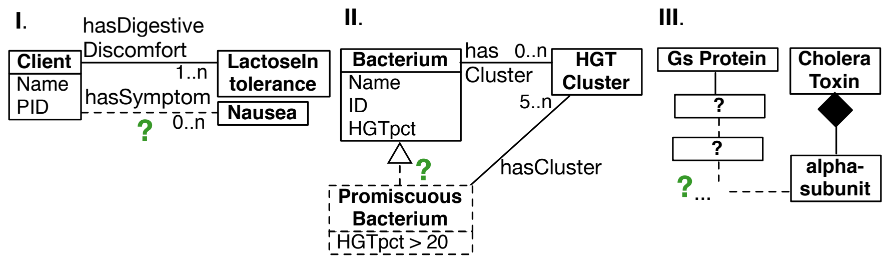


Figure 1.7 Graphical depictions of the three query patterns to find 'new' classes or relationships supported by the data; (i): correlation; (ii): hypothesis about existence of subclass PromiscuousBacterium; (iii): path query to check whether the Gs protein somehow relates to the alpha-subunit of the CholeraToxin.

standard knowledge bases [43]. For brevity, let lower-case letters denote instances of their respective classes in the conceptual data model, then we have:

i. "for each $X(x)$, $Y(y)$, $R(r)$, $R(X,Y)$, does there exist a $Z(z)$, $S(s)$, such that there exists ≥ 1 x and $s(x,z)$?". This is a query for the 'known unknown' Z.
ii. "for each $X(x)$, $Y(y)$, $R(r)$, $R(X,Y)$ in the data store, does there exist an $s(x,z)$ and an $t(x,a)$ where $Z(z)$, $S(s)$, $A(a)$, $T(t)$ hold?" This tests the hypothesis that there may be a quaternary relationship among A, X, Y, and Z instead of three binaries, or perhaps a subtype X' that satisfies the conditions.
iii. A path query (arbitrary relatedness): "for each $X(x)$, return any $r_1, ...r_n$, their type of role and the concepts $Y_1, ..., Y_n$ they are related to".

The first pattern is illustrated in Figure 1.7-I in the setting of electronic health records where X = Client, S = has symptom, and Z = Nausea, thereby querying if patients suffering from lactose intolerance have the symptom of being nauseous.

The aim in the third pattern is to find type-level knowledge where the query answer also includes the class the instance(s) belong to. This still can be tractable if one considers only classes directly related to X, but exploring the search space of sequences of conjunctive queries of arbitrary length is unrealistic. Restricted queries have been examined regarding discovering the relationships between Histone code, DNA sequence, and Gene expression regulation [57]; an example is depicted in Figure 1.7-III, where one queries for a path that connects the G_s protein to the α-subunit only.

An unconstrained type (ii) query leads to a combinatorial explosion, which can be contained by requiring that the user selects several classes and relationships when composing the query, therewith keeping a degree of knowledge discovery; e.g., to find the type (species) of a plant specimen and to refine a classification of enzymes by adding properties. This type of query is not yet supported by standard DL and OWL reasoners, but some cues to implement this can be gleaned from database reverse engineering, whose algorithms detect concepts, relations, and mandatory and uniqueness constraints from the table definitions and data in the tables. A variation on this theme is to consider also incomplete information with rough sets, which has been shown to be useful in hypothesis testing [39]. For instance, it is known that some bacteria transfer more genes horizontally than others do (‘promiscuous bacteria’), but is it not clear what the characteristics are and who has them. One may hypothesize that promiscuous bacteria have, say, $> 20\%$ of their genes that are predicted to be horizontally acquired and have ≥ 5 clusters of horizontally transferred genes (see Figure 1.7-II): in $\mathcal{CM}_{com}$, we have $\textsc{att}(\texttt{PromiscuousBacterium}) = \{\texttt{HGTpct} : \texttt{Real}_{>20}\}$, $\textsc{rel}(\texttt{hasClusters}) = \{\texttt{org} : \texttt{PromiscuousBacterium}, \texttt{geneclust} : \texttt{HGTCluster}\}$, and the constraint $\textsc{card}(\texttt{PromiscuousBacterium}, \texttt{HGTCluster}, \texttt{hasCluster}) = 5..\texttt{n}$. The data retrieved will be a set of bacteria of which some have the same values for the chosen properties, yet they are assumed to be distinct bacteria. Where possible, one can add new properties in successive steps to find the right combination of properties and thereby have discovered the means to indeed distinguish the bacteria. Any potentially useful intermediate combinations of properties easily can be included in the conceptual data model and be subjected to automated classification as was illustrated in Example 6.

1.5 CONCLUSIONS AND OUTLOOK

Ontology-driven formal conceptual data modeling brings rigor to database and software development in the form of ontological guidance and use of ontologies to represent information more accurately, therewith resulting in a better quality conceptual data model, hence, better software. The formal foundation ensures precision in representation of the semantics of the subject domain and enables automated reasoning over conceptual data models, which not only detects inconsistencies and derives implicit information and thereby contributes to the quality of conceptual data models, but also can be used in biological knowledge discovery processes. The latter include services such as consistency checking, class and instance classification, and querying. To substantiate these advantages, we presented a formal foundation for UML, EER, and ORM, being $\mathcal{CM}_{com}$, which has an equi-satisfiable $\mathcal{DLR}_{ifd}$ knowledge base. Ontological guidance to motivate better modeling choices was illustrated with a refinement for representing catalytic reactions. There are many language features and extensions thereof. We demonstrated several claimed to be ‘non-representable’ biological knowledge actually can be represented in $\mathcal{CM}_{com}$ (hence, also in $\mathcal{DLR}_{ifd}$), such as constraints among relationships and that for other requirements language extensions do exist that can give a formal semantics to, among others, temporal knowledge. The latter was demonstrated with transforming entities and related processes in a cascade of interactions of viral infection. Automated reasoning services were illustrated for taxonomic classification, and three different query patterns to find new type-level information were described.

Ontology-driven formal conceptual data modeling is still a relatively young field, and many more usage scenarios are yet to be investigated fully, such as handling incomplete information in hypothesis testing [39], how OntoClean [28] ideas can be incorporated in conceptual data modeling methodologies, and developing a formal link between ontologies and conceptual data models. Development of CASE tools with both a unifying formalism—be this $\mathcal{CM}_{com}$ or another language—and an integrated automated reasoner is necessary as well, not just with one language interface [22] but with multiple graphical syntaxes. Temporal reasoning beyond $\mathcal{ER}_{VT}$ and its $\mathcal{DLR}_{\mathcal{US}}$ foundation, principally either as extension to UML Class

Diagrams or as formalization of Sequence and Activity Diagrams, also may yield useful results for biological knowledge discovery.

REFERENCES

1. A. Artale, D. Calvanese, R. Kontchakov, V. Ryzhikov, and M. Zakharyaschev. Reasoning over extended ER models. In *ER-07*, volume 4801 of *LNCS*, pages 277–292. Springer, 2007.
2. A. Artale, E. Franconi, F. Wolter, and M. Zakharyaschev. A temporal description logic for reasoning about conceptual schemas and queries. In S. Flesca, S. Greco, N. Leone, and G. Ianni, editors, *Proceedings of the 8th Joint European Conference on Logics in Artificial Intelligence (JELIA-02)*, volume 2424 of *LNAI*, pages 98–110. Springer Verlag, 2002.
3. A. Artale, N. Guarino, and C. M. Keet. Formalising temporal constraints on part-whole relations. In G. Brewka and J. Lang, editors, *11th International Conference on Principles of Knowledge Representation and Reasoning (KR'08)*, pages 673–683. AAAI Press, 2008. Sydney, Australia, September 16-19, 2008.
4. A. Artale and C. M. Keet. Essential, mandatory, and shared parts in conceptual data models. In T. Halpin, H. Proper, and J. Krogstie, editors, *Innovations in Information Systems modeling: Methods and Best Practices*, Advances in Database Research Series, pages 17–52. IGI Global, 2008.
5. A. Artale, C. Parent, and S. Spaccapietra. Evolving objects in temporal information systems. *Annals of Mathematics and Artificial Intelligence*, 50(1-2):5–38, 2007.
6. F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logics Handbook – Theory and Applications.* Cambridge University Press, 2 edition, 2008.
7. S. Bandini and A. Mosca. Mereological knowledge representation for the chemical formulation. In *2nd Workshop on Formal Ontologies Meets Industry 2006 (FOMI2006)*, pages 55–69, Trento, Italy, December 2006.
8. E. Beisswanger, S. Schulz, H. Stenzhorn, and U. Hahn. BioTop: An upper domain ontology for the life sciences - a description of its current structure, contents, and interfaces to OBO ontologies. *Applied Ontology*, 3(4):205–212, 2008.
9. D. Berardi, D. Calvanese, and G. De Giacomo. Reasoning on UML class diagrams. *Artificial Intelligence*, 168(1-2):70–118, 2005.
10. BFO. Basic formal ontology, (last accessed August 2010). http://www.ifomis.org/bfo.
11. E. Bornberg-Bauer and N. Paton. Conceptual data modelling for bioinformatics. *Briefings in Bioinformatics*, 3(2):166180, 2002.
12. D. Calvanese and G. De Giacomo. *The DL Handbook: Theory, Implementation and Applications*, chapter Expressive description logics, pages 178–218. Cambridge University Press, 2003.
13. D. Calvanese, G. De Giacomo, and M. Lenzerini. Identification constraints and functional dependencies in description logics. In *Proc. of the 17th Int. Joint Conf. on Artificial Intelligence (IJCAI 2001)*, pages 155–160, 2001.
14. D. Calvanese, G. D. Giacomo, D. Lembo, M. Lenzerini, A. Poggi, M. Rodríguez-Muro, and R. Rosati. Ontologies and databases: The DL-Lite approach. In S. Tessaris and E. Franconi, editors, *Semantic Technologies for Informations Systems - 5th Int.*

Reasoning Web Summer School (RW 2009), volume 5689 of *LNCS*, pages 255–356. Springer, 2009.

15. D. Calvanese, C. M. Keet, W. Nutt, M. Rodríguez-Muro, and G. Stefanoni. Web-based graphical querying of databases through an ontology: the WONDER system. In *Proceedings of ACM Symposium on Applied Computing (ACM SAC'10)*, pages 1389–1396. ACM, 2010. March 22-26 2010, Sierre, Switzerland.
16. D. Calvanese, M. Lenzerini, and D. Nardi. *Logics for Databases and Information Systems*, chapter Description logics for conceptual data modeling. Kluwer, Amsterdam, 1998.
17. D. Calvanese, M. Lenzerini, and D. Nardi. Unifying class-based representation formalisms. *Journal of Artificial Intelligence Research*, 11:199–240, 1999.
18. P. P. Chen. The entity-relationship model—toward a unified view of data. *ACM Transactions on Database Systems*, 1(1):9–36, 1976.
19. E. Demir et al. The BioPAX community standard for pathway data sharing. *Nature Biotechnology*, 28(9):935–942, 2010.
20. H. El-Ghalayini, M. Odeh, R. McClatchey, and D. Arnold. Deriving conceptual data models from domain ontologies for bioinformatics. In *2nd Conference on Information and Communication Technologies (ICTTA'06)*, pages 3562 – 3567. IEEE Computer Society, 2006.
21. R. Elmasri, F. Ji, and J. Fu. Modeling biomedical data. In J. Chen and E. Amandeep S. Sidhu, editors, *Biological database modeling*, chapter 3. Artech House Publishers, 2007.
22. P. R. Fillottrani, E. Franconi, and S. Tessaris. The ICOM 3.0 intelligent conceptual modelling tool and methodology. *Semantic Web Journal*, in print, 2011.
23. S. Garcia-Vallvé, E. Guzman, M. Montero, and A. Romeu. HGT-DB: a database of putative horizontally transferred genes in prokaryotic complete genomes. *Nucleic Acids Research*, 31(1):187–189, 2003.
24. Gene Ontology Consortium. Gene Ontology: tool for the unification of biology. *Nature Genetics*, 25:25–29, 2000.
25. N. Guarino. Formal ontology and information systems. In *Proceedings of Formal Ontology in Information Systems (FOIS'98)*. Amsterdam: IOS Press, 1998.
26. N. Guarino. The ontological level: Revisiting 30 years of knowledge representation. In A. Borgida et al., editors, *Mylopoulos Festschrift*, volume 5600 of *LNCS*, pages 52–67. Springer, 2009.
27. N. Guarino and G. Guizzardi. In the defense of ontological foundations for conceptual modeling. *Scandinavian Journal of Information Systems*, 18(1):(debate forum, 9p), 2006.
28. N. Guarino and C. Welty. An overview of OntoClean. In S. Staab and R. Studer, editors, *Handbook on ontologies*, pages 151–159. Springer Verlag, 2004.
29. G. Guizzardi. *Ontological Foundations for Structural Conceptual Models*. Phd thesis, University of Twente, The Netherlands. Telematica Instituut Fundamental Research Series No. 15, 2005.
30. T. Halpin and T. Morgan. *Information modeling and relational databases*. Morgan Kaufmann, 2nd edition, 2008.
31. H. Herre and B. Heller. Semantic foundations of medical information systems based on top-level ontologies. *Knowledge-Based Systems*, 19:107–115, 2006.

32. M. Jarrar, J. Demy, and R. Meersman. On using conceptual data modeling for ontology engineering. *Journal on Data Semantics: Special issue on Best papers from the ER/ODBASE/COOPIS 2002 Conferences*, 1(1):185–207, 2003.

33. T. Kazic. Putting semantics into the semantic web: How well can it capture biology? *Proc. of Pacific Symposium in Biocomputing*, 11:140–151, 2006.

34. C. M. Keet. Biological data and conceptual modelling methods. *Journal of Conceptual Modeling*, 29, October 2003. http://www.inconcept.com/jcm.

35. C. M. Keet. Part-whole relations in object-role models. In R. Meersman, Z. Tari, P. Herrero., and et al., editors, *2nd International Workshop on Object-Role Modelling (ORM 2006), OTM Workshops 2006*, volume 4278 of *LNCS*, pages 1116–1127. Berlin: Springer-Verlag, 2006. Montpellier, France, Nov 2-3, 2006.

36. C. M. Keet. A formal comparison of conceptual data modeling languages. In *13th International Workshop on Exploring Modeling Methods in Systems Analysis and Design (EMMSAD'08)*, volume 337 of *CEUR-WS*, pages 25–39, 2008. 16-17 June 2008, Montpellier, France.

37. C. M. Keet. Constraints for representing transforming entities in bio-ontologies. In R. Serra and R. Cucchiara, editors, *11th Congress of the Italian Association for Artificial Intelligence (AI*IA 2009)*, volume 5883 of *LNAI*, pages 11–20. Springer Verlag, 2009. Reggio Emilia, Italy, Dec. 9-12, 2009.

38. C. M. Keet. Mapping the Object-Role Modeling language ORM2 into Description Logic language $\mathcal{DLR}_{ifd}$. Technical Report arXiv:cs.LO/0702089v2, KRDB Research Centre, Free University of Bozen-Bolzano, Italy, April 2009. arXiv:cs.LO/0702089v2.

39. C. M. Keet. Ontology engineering with rough concepts and instances. In P. Cimiano and H. Pinto, editors, *17th International Conference on Knowledge Engineering and Knowledge Management (EKAW'10)*, volume 6317 of *LNCS*, pages 507–517. Springer, 2010. 11-15 October 2010, Lisbon, Portugal.

40. C. M. Keet. Enhancing identification mechanisms in UML class diagrams with meaningful keys. In *Proceeding of the SAICSIT Annual Research Conference 2011 (SAICSIT'11)*, pages 283–286. ACM Conference Proceedings, 2011. Cape Town, South Africa, October 3-5, 2011.

41. C. M. Keet and A. Artale. Representing and reasoning over a taxonomy of part-whole relations. *Applied Ontology – Special issue on Ontological Foundations for Conceptual Modeling*, 3(1-2):91–110, 2008.

42. C. M. Keet and A. Artale. A basic characterization of relation migration. In R. Meersman et al., editors, *OTM Workshops, 6th International Workshop on Fact-Oriented Modeling (ORM'10)*, volume 6428 of *LNCS*, pages 484–493. Springer, 2010. October 27-29, 2010, Hersonissou, Crete, Greece.

43. C. M. Keet, M. Roos, and M. S. Marshall. A survey of requirements for automated reasoning services for bio-ontologies in OWL. In *Proceedings of the 3rd Workshop on OWL: Experiences and Directions (OWLED 2007)*, volume 258 of *CEUR-WS*, 2007. 6-7 June 2007, Innsbruck, Austria.

44. KEGG. Kyoto Encyclopedia of Genes and Genomes. http://www.genome.jp/kegg/.

45. T. Lukasiewicz and U. Straccia. Managing uncertainty and vagueness in description logics for the semantic web. *Journal of Web Semantics*, 6(4):291–308, 2008.

46. C. Lutz, D. Toman, and F. Wolter. Conjunctive query answering in the description logic EL using a relational database system. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence IJCAI'09.* AAAI Press, 2009.

47. Z. Ma, F. Zhang, L. Yan, and J. Cheng. Representing and reasoning on fuzzy UML models: A description logic approach. *Expert Systems with Applications*, 38(3):2536–2549, 2011.

48. J. S. Madin, S. Bowers, M. P. Schildhauer, and M. B. Jones. Advancing ecological research with ontologies. *Trends in Ecology & Evolution*, 23(3):159–168, 2008.

49. C. Masolo, S. Borgo, A. Gangemi, N. Guarino, and A. Oltramari. Ontology library. WonderWeb Deliverable D18 (ver. 1.0)., 2003. http://wonderweb.semanticweb.org.

50. B. Motik, P. F. Patel-Schneider, and B. Parsia. OWL 2 web ontology language structural specification and functional-style syntax. W3c recommendation, W3C, 27 Oct. 2009. http://www.w3.org/TR/owl2-syntax/.

51. D. A. Natale et al. The Protein Ontology: a structured representation of protein forms and complexes. *Nucleic Acids Research*, 39(Database issue):D539–D545, 2011.

52. Object Management Group. Superstructure specification. Standard 2.3, Object Management Group, May 2010. http://www.omg.org/spec/UML/2.3/.

53. C. Parent, S. Spaccapietra, and E. Zimányi. *Conceptual modeling for traditional and spatio-temporal applications—the MADS approach.* Berlin Heidelberg: Springer Verlag, 2006.

54. O. Pastor, A. M. Levin, J. C. Casamayor, M. Celma, L. E. Eraso, M. J. Villanueva, and M. Perez-Alonso. Enforcing conceptual modeling to improve the understanding of human genome. In *Fourth International Conference on Research Challenges in Information Science (RCIS'10)*, pages 85–92. IEEE Computer Society, 2010. Nice, France, 19-21 May 2010.

55. A. Queralt and E. Teniente. Reasoning on UML class diagrams with OCL constraints. In D. Embley, A. Olivé, and S. Ram, editors, *Proceedings of ER'06*, volume 4215 of *LNCS*, pages 497–512. Springer-Verlag, 2006.

56. A. Queralt and E. Teniente. Decidable reasoning in UML schemas with constraints. In Z. Bellahsene and M. Léonard, editors, *CAiSE*, volume 5074 of *Lecture Notes in Computer Science*, pages 281–295. Springer, 2008.

57. M. Roos, H. Rauwerda, M. Marshall, L. Post, M. Inda, C. Henkel, and T. Breit. Towards a virtual laboratory for integrative bioinformatics research. In C. M. Keet and E. Franconi, editors, *CSBio Reader: Extended abstracts of CS & IT with/for Biology Seminar Series 2005*, pages 18–25. Free University of Bozen-Bolzano, 2005.

58. D. Shegogue and W. J. Zheng. Object-oriented biological system integration: a SARS coronavirus example. *Bioinformatics*, 21(10):2502–2509, 2005.

59. B. Smith, M. Ashburner, C. Rosse, J. Bard, W. Bug, W. Ceusters, L. Goldberg, K. Eilbeck, A. Ireland, C. Mungall, T. OBI Consortium, N. Leontis, A. Rocca-Serra, A. Ruttenberg, S.-A. Sansone, M. Shah, P. Whetzel, and S. Lewis. The OBO Foundry: Coordinated evolution of ontologies to support biomedical data integration. *Nature Biotechnology*, 25(11):1251–1255, 2007.

60. B. Smith, W. Ceusters, B. Klagges, J. Köhler, A. Kumar, J. Lomax, C. Mungall, F. Neuhaus, A. L. Rector, and C. Rosse. Relations in biomedical ontologies. *Genome Biology*, 6:R46, 2005.

61. V. Sugumaran and V. C. Storey. The role of domain ontologies in database design: An ontology management and conceptual modeling environment. *ACM Transactions on Database Systems*, 31(3):1064–1094, 2006.

62. The UniProt Consortium. Ongoing and future developments at the universal protein resource. *Nucleic Acids Res.*, 39:D214–D219, 2011.

63. K. Wolstencroft, R. Stevens, and V. Haarslev. Applying OWL reasoning to genomic data. In C. Baker and H. Cheung, editors, *Semantic Web: revolutionizing knowledge discovery in the life sciences*, pages 225–248. Springer: New York, 2007.