# Can OWL model football leagues?

Diego Calvanese[1], Giuseppe De Giacomo[2], Domenico Lembo[2],
Maurizio Lenzerini[2], Riccardo Rosati[2]

[1] Faculty of Computer Science
Free University of Bozen-Bolzano
Piazza Domenicani 3
I-39100 Bolzano, Italy
calvanese@inf.unibz.it

[2] Dip. di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Salaria 113
I-00198 Roma, Italy
*lastname*@dis.uniroma1.it

**Abstract.** Identity is one of the main principles in ontology engineering. Mechanisms to identify objects by using attribute values or by their participation to relationships with other objects are present in most conceptual modeling formalisms used in software engineering and database or information system design. Identification mechanisms have also been investigated in expressive Description Logics. However they are currently missing in OWL. In this paper we argue for their usefulness, and we show how to extend *DL-Lite*$_A$, a tractable fragment of OWL, with identification assertions without losing tractability of reasoning.

## 1 Introduction

Identity is one of the main principles in ontology engineering, and methods for explicitly talking about identity are provided in some modeling languages. For example, mechanisms for identifying objects using attribute values or their participation to relationships with other objects are present in most conceptual modeling formalisms used in software engineering and database or information system design [7, 6, 8].

On the other hand, in OWL [2][1], the only way to specify identification is through the use of one-to-one relationships, and this corresponds to a very limited form of identification. More powerful identification mechanisms have been already studied in the context of very expressive Description Logics (DLs), see, e.g., [5, 10, 11], but they have not been incorporated in OWL yet.

We argue for the usefulness of identification assertions in modeling a domain of interest through an ontology. For example, in the context of geografic information systems, the fact that a location is identified by its cooordinates should be considered as part of the very definition of location. However, this cannot be expressed in OWL. The lack of identification mechanisms is even more serious if one considers that in OWL both $n$*-ary relations* and *attributes of roles* are missing. Indeed, the only way to represent an arbitrary $n$-ary relation in OWL is through the well-known *reification* technique [1]. Notice that, by reification we mean the use of an object to denote a tuple, and it should not be confused with reification where an object is used to denote a predicate, as happens, e.g., in RDF. Reification, as intended in this paper, actually needs identification assertions, as we briefly illustrate in the following simple example. Consider the notion

[1] Although we generically use the term "OWL", in this paper we focus on OWL-DL.

of exam, where each instance relates a student, a course, and a grade. Intuitively, this notion can be modeled in First Order Logic (FOL) with a ternary relation. In OWL, such a relation can be represented by the reified concept $C_{exam}$, two binary roles $R_{student}$ and $R_{course}$, and an attribute $A_{grade}$ on concept $C_{exam}$. However, in order for this reified representation to be correct, we must also impose that no two distinct instances of $C_{exam}$ exist that are connected to the same pair of fillers for $R_{student}$ and $R_{course}$. This is exactly what an identification assertion can be used for.

In this paper we discuss and illustrate the need of identification mechanisms in detail. We use as running example a simple OWL ontology about European Football Leagues, asking ourself whether we can formally represent in OWL the knowledge that is commonly shared on how the various leagues and their matches are organized. We argue that identification assertions are needed for accurately modeling such a domain. Then, we propose a mechanism for specifying and reasoning on identification assertions in a tractable DL, namely *DL-Lite*$_A$ [3], corresponding to a particularly well behaved fragment of OWL. It turns out that identification assertions can indeed be added to this fragment without falling into intractability of both TBox reasoning and query answering.

## 2 Motivating example

Our goal in this section is to illustrate the need of identification assertions by means of an example. We aim at defining suitable concepts and relationships for modeling the annual national football[2] championships in Europe. The championship for a specific year and for a specific nation is called *league* (e.g., the 2006 Spanish Liga). A league is structured in terms of a set of *rounds*. In every round, a set of *matches* take place. Each match is played by one *home team* and one *host team*, and is umpired by one *referee*. A match is played in a specific date, and every match that has been played is characterized by its result, where the result is given in terms of the number of goals scored by the home team and the host team. Note that different matches scheduled for the same round may be played in different dates.

In Figure 1, we show a diagrammatic representation of the ontology for the above described domain. Concepts are represented as ovals, attributes are represented as squares connected to the concepts they refer to, ISA between concepts is represented by an arrow, and roles are drawn as lines connecting the appropriate concepts. The same ontology expressed in OWL is given in the Appendix.

One might wonder whether this ontology faithfully describes the domain of interest. Indeed, an ontology should select as accurately as possible the class of intended models that describe the application domain. So a natural question to ask is how accurate the above ontology is in representing the semantics of our football domain. It is not hard to see that the OWL ontology described above fails to model the following aspects:

1. no two leagues with the same year and the same nation exist;
2. within a certain league, the code associated to a round is unique;
3. every match is identified by its code within its round;

[2] Football is called "soccer" in the United States.

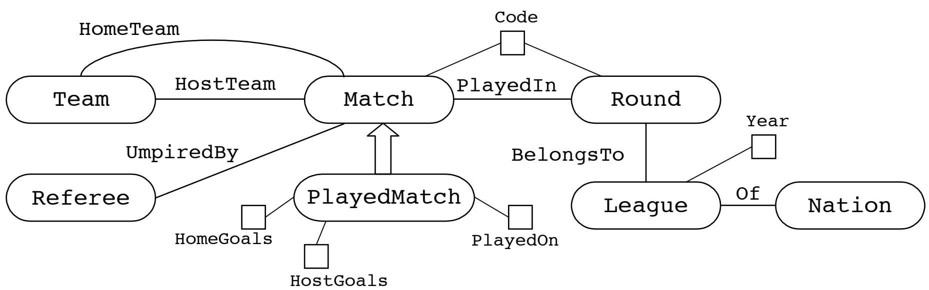


**Fig. 1.** Diagramatic representation of the football ontology

4. every referee can umpire at most one match in the same round;
5. no team can be the home team of more than one match per round;
6. no team can be the host team of more than one match per round.

Unfortunately, with the modeling constructs available in OWL, the above characteristics simply cannot be expressed. Indeed, all such characteristics requires the notion of *identifier*, which is missing in OWL.

## 3 Adding identification assertions to the description logic *DL-Lite*$_A$

In this section, we consider the logic *DL-Lite*$_A$ [3], and we extend its language with indentification assertions. As usual in DLs, *DL-Lite*$_A$ allows one to represent the universe of discourse in terms of concepts, denoting sets of objects, and roles, denoting binary relations between objects. In addition, *DL-Lite*$_A$ allows one to use value-domains, a.k.a. concrete domains [9], denoting unbounded sets of (data) values, and concept attributes, denoting binary relations between objects and values[3]. In particular, the value-domains that we consider here are those corresponding to unbounded (i.e., value-domains with an unbounded size) RDF data types, such as integers, real, strings, etc.

To extend *DL-Lite*$_A$ with identification assertions, we start by adding such assertions to the DL *DL-Lite*$_{FR}$, which combines the main features of two DLs presented in [4], called *DL-Lite*$_F$ and *DL-Lite*$_R$, respectively. We use the following notation:

- $A$ denotes an *atomic concept*, $B$ a *basic concept*, $C$ a *general concept*, and $\top_C$ the *universal concept*;
- $E$ denotes a basic value-domain, i.e., the range of an attribute, $T_1, \ldots, T_n$ denote the $n$ pairwise disjoint unbounded RDF data types used in our logic, and $F$ denotes a *general value-domain*, which can be either an unbounded RDF data type $T_i$ or the *universal value-domain* $\top_D$;
- $P$ denotes an *atomic role*, $Q$ a *basic role*, and $R$ a *general role*;
- $U_C$ denotes an *atomic attribute*, and $V_C$ a *general attribute*.

[3] The logic discussed in [3] is actually more expressive than *DL-Lite*$_A$, since it includes role attributes, user-defined domains, as well as inclusion assertions over such domains.

Given an attribute $U_C$, we call the *domain* of $U_C$, denoted by $\delta(U_C)$, the set of objects that $U_C$ relates to values, and we call *range* of $U_C$, denoted by $\rho(U_C)$, the set of values related to objects by $U_C$.

We are now ready to define *DL-Lite*$_{FR}$ expressions as follows.

– Basic and general concept expressions:

$$\begin{array}{rcl} B & ::= & A \mid \exists Q \mid \delta(U_C) \\ C & ::= & \top_C \mid B \mid \neg B \mid \exists Q.C \end{array}$$

– Basic and general value-domain expressions:

$$\begin{array}{rcl} E & ::= & \rho(U_C) \\ F & ::= & \top_D \mid T_1 \mid \cdots \mid T_n \end{array}$$

– Attribute expressions:

$$V_C ::= U_C \mid \neg U_C$$

– Basic and general role expressions:

$$\begin{array}{rcl} Q & ::= & P \mid P^- \\ R & ::= & Q \mid \neg Q \end{array}$$

A *DL-Lite*$_{FR}$ *knowledge base* (KB) with identification assertions $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is constituted by two components: a TBox $\mathcal{T}$, used to represent intensional knowledge, and an ABox $\mathcal{A}$, used to represent extensional knowledge. A *DL-Lite*$_{FR}$ *TBox* with identification assertions is constituted by a finite set of assertions of the following forms:

– Inclusion assertions:

$$\begin{array}{cl} B \sqsubseteq C & \textit{concept inclusion assertion} \\ Q \sqsubseteq R & \textit{role inclusion assertion} \\ E \sqsubseteq F & \textit{value-domain inclusion assertion} \\ U_C \sqsubseteq V_C & \textit{attribute inclusion assertion} \end{array}$$

A concept inclusion assertion expresses that a (basic) concept $B$ is subsumed by a (general) concept $C$. Analogously for the other types of inclusion assertions.

– Functionality assertions on atomic attributes or basic roles:

$$(\mathsf{funct}\ I) \qquad \textit{functionality assertion}$$

where $I$ denotes either an atomic attribute or a basic role. A functionality assertion expresses the (global) functionality of an atomic attribute or a basic role.

– Identification assertions:

$$(\mathsf{id}\ B\ I_1, \ldots, I_n) \qquad \textit{identification assertion}$$

where $B$ denotes a basic concept and each $I_j$ denotes either an atomic attribute or a basic role. Such an assertion specifies that the combination of properties $I_1, \ldots, I_n$ identifies the instances of the basic concept $B$. More precisely, it imposes that two instances of $B$ cannot agree on all the fillers for $I_1, \ldots, I_n$. We call $I_1, \ldots, I_n$ the *identification list* of the identification assertion.

As for the ABox, we introduce two disjoint alphabets, called $\Gamma_O$ and $\Gamma_V$, respectively. Symbols in $\Gamma_O$, called *object constants*, are used to denote objects, while symbols in $\Gamma_V$, called *value constants*, are used to denote data values. A *DL-Lite$_{FR}$ ABox* is a finite set of assertions of the form:

$$A(a), \quad P(a,b), \quad U_C(a,c) \qquad \textit{membership assertions}$$

where $a$ and $b$ are constants in $\Gamma_O$, and $c$ is a constant in $\Gamma_V$.

The semantics of *DL-Lite$_{FR}$* with identification assertions is given in terms of FOL interpretations. An *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a first order structure over the interpretation domain $\Delta^{\mathcal{I}}$ that is the disjoint union of $\Delta_O^{\mathcal{I}}$ and $\Delta_V^{\mathcal{I}}$, with an *interpretation function* $\cdot^{\mathcal{I}}$ such that

- for all $a \in \Gamma_O$, we have that $a^{\mathcal{I}} \in \Delta_O^{\mathcal{I}}$;
- for all $c \in \Gamma_V$, we have that $c^{\mathcal{I}} \in \Delta_V^{\mathcal{I}}$;
- for all $c, d \in \Gamma$, we have that $c \neq d$ implies $c^{\mathcal{I}} \neq d^{\mathcal{I}}$;
- and the following conditions are satisfied:

$$\begin{array}{ll}
\top_C^{\mathcal{I}} = \Delta_O^{\mathcal{I}} & (P^-)^{\mathcal{I}} = \{ (o,o') \mid (o',o) \in P^{\mathcal{I}} \} \\
\top_D^{\mathcal{I}} = T_1^{\mathcal{I}} \oplus \ldots \oplus T_n^{\mathcal{I}} = \Delta_V^{\mathcal{I}} & (\rho(U_C))^{\mathcal{I}} = \{ v \mid \exists o.\, (o,v) \in U_C^{\mathcal{I}} \} \\
A^{\mathcal{I}} \subseteq \Delta_O^{\mathcal{I}} & (\delta(U_C))^{\mathcal{I}} = \{ o \mid \exists o.\, (o,v) \in U_C^{\mathcal{I}} \} \\
P^{\mathcal{I}} \subseteq \Delta_O^{\mathcal{I}} \times \Delta_O^{\mathcal{I}} & (\exists Q)^{\mathcal{I}} = \{ o \mid \exists o'.\, (o,o') \in Q^{\mathcal{I}} \} \\
U_C^{\mathcal{I}} \subseteq \Delta_O^{\mathcal{I}} \times \Delta_V^{\mathcal{I}} & (\exists Q.C)^{\mathcal{I}} = \{ o \mid \exists o'.\, (o,o') \in Q^{\mathcal{I}} \wedge o' \in C^{\mathcal{I}} \} \\
(\neg U_C)^{\mathcal{I}} = (\Delta_O^{\mathcal{I}} \times \Delta_V^{\mathcal{I}}) \setminus U_C^{\mathcal{I}} & (\neg B)^{\mathcal{I}} = \Delta_O^{\mathcal{I}} \setminus B^{\mathcal{I}} \\
(\neg Q)^{\mathcal{I}} = (\Delta_O^{\mathcal{I}} \times \Delta_O^{\mathcal{I}}) \setminus Q^{\mathcal{I}} &
\end{array}$$

We define when an interpretation $\mathcal{I}$ *satisfies* an assertion (i.e., is a *model* of the assertion) as follows (below, each $e$ and $f$, possibly with subscript, is an element of either $\Delta_O^{\mathcal{I}}$ or $\Delta_V^{\mathcal{I}}$, depending on the context, each $t$, possibly with subscript, is a constant of either $\Gamma_O$ or $\Gamma_V$, depending on the context, $a$ and $b$ are constants in $\Gamma_O$, and $c$ is a constant in $\Gamma_V$). Specifically, an interpretation $\mathcal{I}$ *satisfies*:

- an inclusion assertion $\alpha \sqsubseteq \beta$, if $\alpha^{\mathcal{I}} \subseteq \beta^{\mathcal{I}}$;
- a functionality assertion $(\mathsf{funct}\ \gamma)$, where $\gamma$ is either $P$, $P^-$, or $U_C$, if, for each $e_1, e_2, e_3$, we have that $(e_1, e_2) \in \gamma^{\mathcal{I}}$ and $(e_1, e_3) \in \gamma^{\mathcal{I}}$ implies $e_2 = e_3$;
- an identication assertion $(\mathsf{id}\ B\ I_1, \ldots, I_n)$, if for all $e_1, e_2 \in B^{\mathcal{I}}$ and for all $f_1^1, \ldots, f_1^n, f_2^1, \ldots, f_2^n$, we have that $(e_1, f_1^j) \in I_j^{\mathcal{I}}$ and $(e_2, f_2^j) \in I_j^{\mathcal{I}}$, for $j \in \{1, \ldots, n\}$, implies $e_1 = e_2$;
- a membership assertion $A(t)$, if $t^{\mathcal{I}} \in A^{\mathcal{I}}$;
- a membership assertion $\beta(t_1, t_2)$, where $\beta$ is either $P$ or $U_C$, if $(t_1^{\mathcal{I}}, t_2^{\mathcal{I}}) \in \beta^{\mathcal{I}}$.

A *model of a KB* $\mathcal{K}$ is an interpretation $\mathcal{I}$ that is a model of all assertions in $\mathcal{K}$. A KB is *satisfiable* if it has at least one model. A KB $\mathcal{K}$ *logically implies* an assertion $\alpha$ if all models of $\mathcal{K}$ are also models of $\alpha$.

An atomic attribute $U_C$ (resp. a basic role $Q$) is called an *identifying property in* $\mathcal{T}$, if $\mathcal{T}$ contains a functionality assertion $(\mathsf{funct}\ U_C)$ (resp. $(\mathsf{funct}\ Q)$ or $(\mathsf{funct}\ Q^-)$), or $\mathcal{T}$ contains an indentification assertion whose identification list includes $U_C$ (resp. $Q$ or $Q^-$). Also, an atomic attribute or a basic role is called *primitive in* $\mathcal{T}$, if it does not appear positively in the right-hand side of an inclusion assertion of $\mathcal{T}$, and it does not appear in an expression of the form $\exists Q.C$ in $\mathcal{T}$.

We are now ready to define the logic that extends *DL-Lite*$_A$ with identification assertions.

> *A DL-Lite*$_A$ *knowledge base with identification assertions is a pair* $\langle \mathcal{T}, \mathcal{A} \rangle$*, where* $\mathcal{A}$ *is a DL-Lite*$_{FR}$ *ABox, and* $\mathcal{T}$ *is a DL-Lite*$_{FR}$ *TBox with identification assertions such that* $\mathcal{T}$ *satisfies the condition that every identifying property is primitive in* $\mathcal{T}$*.*

Roughly speaking, in our logic, *identifying properties cannot be specialized*, i.e., they cannot be used positively in the right-hand side of inclusion assertions.

One of the crucial properties of *DL-Lite*$_A$ without identification assertions is tractability: TBox reasoning is PTIME and query answering is LOGSPACE in the size of the ABox (indeed SQL reducible) [3]. So one might wonder whether adding identification assertions breaks such a nice behavior. It turns out that identification assertions are indeed harmless w.r.t. tractability of reasoning, in the sense that the complexity bounds above are preserved.

## 4 *DL-Lite*$_A$ with identification assertions: an example

In this section, we consider again the example of Section 2 and encode it in *DL-Lite*$_A$ extended with identification assertions. By virtue of the possibility of specifying identification constraints, we can now provide a more accurate representation of the semantics of our football domain. Indeed, besides translating in *DL-Lite*$_A$ the OWL ontology given in Section 2, which can be done using only *DL-Lite*$_A$ inclusion and functionality assertions, we are also able, through the use of identification assertions, to express all domain aspects listed in Section 2 and missed in the OWL ontology. The resulting *DL-Lite*$_A$ TBox with identification assertions is given in Figure 2.

It is easy to check that every identifying property in the TBox $\mathcal{T}_{ex}$ given in Figure 2 is primitive in $\mathcal{T}_{ex}$, and therefore the TBox is indeed a *DL-Lite*$_A$ TBox with identification assertions. It is also easy to see that:

1. (id *league* $OF$, $\mathbf{year}$) models the property that no two leagues with the same year and the same nation exist;
2. (id *round BELONGS-TO*, $\mathbf{code}$) models the property that the code associated to a round is unique within the league to which the round belongs;
3. (id *match PLAYED-IN*, $\mathbf{code}$) models the property that every match is identified by its code within its round;
4. (id *match UMPIRED-BY*, *PLAYED-IN*) models the property that every referee can umpire at most one match in the same round;
5. (id *match HOME-TEAM*, *PLAYED-IN*) models the property that no team can be the home team of more than one match per round;
6. (id *match HOST-TEAM*, *PLAYED-IN*) models the property that no team can be the host team of more than one match per round.

| *Alphabet* | | | |
|---|---|---|---|
| Atomic Concepts | *league*<br>*nation*<br>*round*<br>*match*<br>*playedMatch*<br>*team*<br>*referee* | Atomic Roles | *BELONGS-TO*<br>*OF*<br>*PLAYED-IN*<br>*HOME-TEAM*<br>*HOST-TEAM*<br>*UMPIRED-BY* |
| Atomic Attributes | **year**<br>**code**<br>**playedOn**<br>**homeGoals**<br>**hostGoals** | *rdfUnboundedDataType* | `xsd:positiveInteger`<br>`xsd:date`<br>`xsd:nonNegativeInteger` |

| *Inclusion Assertions* | |
|---|---|
| $league \sqsubseteq \exists OF$ | $match \sqsubseteq \exists$*UMPIRED-BY* |
| $\exists OF \sqsubseteq league$ | $\exists$*UMPIRED-BY* $\sqsubseteq$ *match* |
| $\exists OF^{-} \sqsubseteq nation$ | $\exists$*UMPIRED-BY*$^{-}$ $\sqsubseteq$ *referee* |
| *round* $\sqsubseteq \exists$*BELONGS-TO* | *playedMatch* $\sqsubseteq$ *match* |
| $\exists$*BELONGS-TO* $\sqsubseteq$ *round* | *match* $\sqsubseteq \delta(\mathbf{code})$ |
| $\exists$*BELONGS-TO*$^{-}$ $\sqsubseteq$ *league* | *round* $\sqsubseteq \delta(\mathbf{code})$ |
| *match* $\sqsubseteq \exists$*PLAYED-IN* | *playedMatch* $\sqsubseteq \delta(\mathbf{playedOn})$ |
| $\exists$*PLAYED-IN* $\sqsubseteq$ *match* | *playedMatch* $\sqsubseteq \delta(\mathbf{homeGoals})$ |
| $\exists$*PLAYED-IN*$^{-}$ $\sqsubseteq$ *round* | *playedMatch* $\sqsubseteq \delta(\mathbf{hostGoals})$ |
| *match* $\sqsubseteq \exists$*HOME-TEAM* | *league* $\sqsubseteq \delta(\mathbf{year})$ |
| $\exists$*HOME-TEAM* $\sqsubseteq$ *match* | $\rho(\mathbf{playedOn}) \sqsubseteq$ `xsd:date` |
| $\exists$*HOME-TEAM*$^{-}$ $\sqsubseteq$ *team* | $\rho(\mathbf{homeGoals}) \sqsubseteq$ `xsd:nonNegativeInteger` |
| *match* $\sqsubseteq \exists$*HOST-TEAM* | $\rho(\mathbf{hostGoals}) \sqsubseteq$ `xsd:nonNegativeInteger` |
| $\exists$*HOST-TEAM* $\sqsubseteq$ *match* | $\rho(\mathbf{code}) \sqsubseteq$ `xsd:positiveInteger` |
| $\exists$*HOST-TEAM*$^{-}$ $\sqsubseteq$ *team* | $\rho(\mathbf{year}) \sqsubseteq$ `xsd:positiveInteger` |

| *Functionality Assertions* | |
|---|---|
| (funct $OF$) | (funct **playedOn**) |
| (funct *BELONGS-TO*) | (funct **homeGoals**) |
| (funct *PLAYED-IN*) | (funct **hostGoals**) |
| (funct *HOST-TEAM*) | (funct **code**) |
| (funct *HOME-TEAM*) | (funct **year**) |
| (funct *UMPIRED-BY*) | |

| *Identification Assertions* | |
|---|---|
| (id *league* $OF$, **year**) | (id *match UMPIRED-BY*, *PLAYED-IN*) |
| (id *round BELONGS-TO*, **code**) | (id *match HOME-TEAM*, *PLAYED-IN*) |
| (id *match PLAYED-IN*, **code**) | (id *match HOST-TEAM*, *PLAYED-IN*) |

**Fig. 2.** The TBox $\mathcal{T}_{ex}$ for the football domain expressed in *DL-Lite*$_A$ with identification assertions

We finally notice that, by reasoning on the TBox $\mathcal{T}_{ex}$, which is in PTIME (see Section 3), we can infer other identification assertions that are logically implied by those asserted in $\mathcal{T}_{ex}$. For example, from the assertions (id *match HOST-TEAM*, *PLAYED-IN*) and *playedMatch* $\sqsubseteq$ *match* we easily get the assertion (id *playedMatch HOST-TEAM*, *PLAYED-IN*), stating that no team can be the host team of more than one played match per round.

## 5 Conclusions

In this paper we have argued for the need of identification mechanisms in ontology languages. We have presented one such mechanism for a tractable fragment of OWL, which turns out to preserve the nice computational properties of the logic.

We are currently investigating more powerful identification assertions with the goal of checking whether they endanger tractability of reasoning. First investigations show that extending our identification assertions with paths rather than simple properties is harmless.


### Acknowledgments

We thank Bernardo Cuenca Grau for helping in the hard task of writing an OWL ontology. This research has been partially supported by the FET project TONES (Thinking ONtologiES), funded by the EU in the 6th Framework Programme under contract number FP6-7603, and by the MIUR FIRB 2005 project "Tecnologie Orientate alla Conoscenza per Aggregazioni di Imprese in Internet" (TOCAI.IT).


## References


1. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2003.
2. S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein. OWL Web Ontology Language reference. W3C Recommendation, Feb. 2004. Available at `http://www.w3.org/TR/owl-ref/`.
3. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, A. Poggi, and R. Rosati. Linking data to ontologies: The description logic DL-Lite$_A$. In *Proc. of OWLED 2006*, 2006.
4. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Data complexity of query answering in description logics. In *Proc. of KR 2006*, pages 260–270, 2006.
5. D. Calvanese, G. De Giacomo, and M. Lenzerini. Identification constraints and functional dependencies in description logics. In *Proc. of IJCAI 2001*, pages 155–160, 2001.
6. P. P. Chen. The Entity-Relationship model: Toward a unified view of data. *ACM Trans. on Database Systems*, 1(1):9–36, Mar. 1976.
7. M. Fowler and K. Scott. *UML Distilled – Applying the Standard Object Modeling Laguage*. Addison Wesley Publ. Co., 1997.
8. R. B. Hull and R. King. Semantic database modelling: Survey, applications and research issues. *ACM Computing Surveys*, 19(3):201–260, Sept. 1987.
9. C. Lutz. Description logics with concrete domains: A survey. In P. Balbiani, N.-Y. Suzuki, F. Wolter, and M. Zakharyaschev, editors, *Advances in Modal Logics*, volume 4. King's College Publications, 2003.
10. C. Lutz, C. Areces, I. Horrocks, and U. Sattler. Keys, nominals, and concrete domains. *J. of Artificial Intelligence Research*, 23:667–726, 2005.
11. D. Toman and G. E. Weddell. On keys and functional dependencies as first-class citizens in description logics. In *Proc. of IJCAR 2006*, pages 647–661, 2006.

## Appendix

We report below the OWL code of the ontology described in Section 2.

```
<!-- Classes -->
  <owl:Class rdf:about="#League">
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#Of"/>
        <owl:someValuesFrom rdf:resource="&owl;Thing"/>
      </owl:Restriction>
    </rdfs:subClassOf>
  </owl:Class>

  <owl:Class rdf:about="#Round">
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#BelongsTo"/>
        <owl:someValuesFrom rdf:resource="&owl;Thing"/>
      </owl:Restriction>
    </rdfs:subClassOf>
  </owl:Class>

  <owl:Class rdf:about="#Match">
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#UmpiredBy"/>
        <owl:someValuesFrom rdf:resource="&owl;Thing"/>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#PlayedIn"/>
        <owl:someValuesFrom rdf:resource="&owl;Thing"/>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#HomeTeam"/>
        <owl:someValuesFrom rdf:resource="&owl;Thing"/>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#HostTeam"/>
        <owl:someValuesFrom rdf:resource="&owl;Thing"/>
      </owl:Restriction>
    </rdfs:subClassOf>
  </owl:Class>

  <owl:Class rdf:about="#Nation"/>

  <owl:Class rdf:about="#PlayedMatch">
    <rdfs:subClassOf rdf:resource="#Match"/>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#HostGoals"/>
        <owl:someValuesFrom rdf:resource="&xsd;anyType"/>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#PlayedOn"/>
        <owl:someValuesFrom rdf:resource="&xsd;anyType"/>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#HomeGoals"/>
        <owl:someValuesFrom rdf:resource="&xsd;anyType"/>
      </owl:Restriction>
    </rdfs:subClassOf>
  </owl:Class>

  <owl:Class rdf:about="#Referee"/>

  <owl:Class rdf:about="#Team"/>
```

```
<!-- Datatypes -->
  <rdfs:Datatype rdf:about="&xsd;anyType">
    <rdf:type rdf:resource="&owl;Thing"/>
  </rdfs:Datatype>

  <rdfs:Datatype rdf:about="&xsd;date">
    <rdf:type rdf:resource="&owl;Thing"/>
  </rdfs:Datatype>

  <rdfs:Datatype rdf:about="&xsd;nonNegativeInteger">
    <rdf:type rdf:resource="&owl;Thing"/>
  </rdfs:Datatype>

  <rdfs:Datatype rdf:about="&xsd;positiveInteger">
    <rdf:type rdf:resource="&owl;Thing"/>
  </rdfs:Datatype>

<!-- Datatype Properties -->
  <owl:DatatypeProperty rdf:about="#Code">
    <rdf:type rdf:resource="&owl;FunctionalProperty"/>
    <rdfs:range rdf:resource="&xsd;positiveInteger"/>
  </owl:DatatypeProperty>

  <owl:DatatypeProperty rdf:about="#HomeGoals">
    <rdf:type rdf:resource="&owl;FunctionalProperty"/>
    <rdfs:range rdf:resource="&xsd;nonNegativeInteger"/>
  </owl:DatatypeProperty>

  <owl:DatatypeProperty rdf:about="#HostGoals">
    <rdf:type rdf:resource="&owl;FunctionalProperty"/>
    <rdfs:range rdf:resource="&xsd;nonNegativeInteger"/>
  </owl:DatatypeProperty>

  <owl:DatatypeProperty rdf:about="#PlayedOn">
    <rdf:type rdf:resource="&owl;FunctionalProperty"/>
    <rdfs:range rdf:resource="&xsd;date"/>
  </owl:DatatypeProperty>

  <owl:DatatypeProperty rdf:about="#Year">
    <rdf:type rdf:resource="&owl;FunctionalProperty"/>
    <rdfs:range rdf:resource="&xsd;positiveInteger"/>
  </owl:DatatypeProperty>

<!-- Object Properties -->
  <owl:FunctionalProperty rdf:about="#BelongsTo">
    <rdf:type rdf:resource="&owl;ObjectProperty"/>
    <rdfs:domain rdf:resource="#Round"/>
    <rdfs:range rdf:resource="#League"/>
  </owl:FunctionalProperty>

  <owl:FunctionalProperty rdf:about="#HomeTeam">
    <rdf:type rdf:resource="&owl;ObjectProperty"/>
    <rdfs:domain rdf:resource="#Match"/>
    <rdfs:range rdf:resource="#Team"/>
  </owl:FunctionalProperty>

  <owl:FunctionalProperty rdf:about="#HostTeam">
    <rdf:type rdf:resource="&owl;ObjectProperty"/>
    <rdfs:domain rdf:resource="#Match"/>
    <rdfs:range rdf:resource="#Team"/>
  </owl:FunctionalProperty>

  <owl:FunctionalProperty rdf:about="#Of">
    <rdf:type rdf:resource="&owl;ObjectProperty"/>
    <rdfs:domain rdf:resource="#League"/>
    <rdfs:range rdf:resource="#Nation"/>
  </owl:FunctionalProperty>

  <owl:FunctionalProperty rdf:about="#PlayedIn">
    <rdf:type rdf:resource="&owl;ObjectProperty"/>
    <rdfs:domain rdf:resource="#Match"/>
    <rdfs:range rdf:resource="#Round"/>
  </owl:FunctionalProperty>

  <owl:FunctionalProperty rdf:about="#UmpiredBy">
    <rdf:type rdf:resource="&owl;ObjectProperty"/>
    <rdfs:domain rdf:resource="#Match"/>
    <rdfs:range rdf:resource="#Referee"/>
  </owl:FunctionalProperty>
```