Ontologies in engineering: The OntoDB/OntoQL platform

Yamine Ait-Ameur · Mickaël Baron · Ladjel Bellatreche · Stéphane Jean · Eric Sardet

Abstract Ontologies have been increasingly used over the past few decades in a wide range of application domains spanning both academic and industrial communities. As ontologies are the cornerstone of the Semantic Web, the technologies developed in this context, including ontology languages, specialized databases and query languages, have become widely used. However, the expressiveness of the proposed ontology languages does not always cover the needs of specific domains. For instance, engineering is a domain for which the LIAS laboratory has proposed dedicated solutions with a worldwide recognition. The underlying assumptions made in the context of the Semantic Web, an open and distributed environment, do not apply to the controlled environments of our projects where the correctness and completeness of modeling can be guaranteed to a certain degree. As a consequence, we have developed over the last decades a specialized standard ontology language named PLIB associated with the OntoDB/OntoQL platform to manage ontological engineering data within a database. The goal of this paper is threefold: (i) to share our experience in manipulating ontologies in the engineering domain by describing their specificities and constraints, (ii) to define a comprehensive classification of ontologies with respect to three main research communities: Artificial Intelligence, Databases and Natural Language Processing and (iii) to present a persistent solution, called OntoDB, for managing extremely large semantic data sets associated with an ontological query language, called OntoQL. These objectives are illustrated by several examples that show the effectiveness and interest of our propositions in several industrial projects in different domains including vehicle manufacturing and CO2 storage.

Yamine Ait-Ameur
IRIT/ ENSHEEIHT, Toulouse, France
E-mail: yamine@enseeiht.fr

Mickal Baron, Ladjel Bellatreche, Stéphane Jean
LIAS/ISAE-ENSMA - University of Poitiers
1, Avenue Clement Ader, 86960 Futuroscope Cedex, France
E-mail: {baron,bellatreche,jean}@ensma.fr

Eric Sardet
CRITT Informatique
Futuroscope Cedex, France
E-mail: {sardet}@critt-informatique.fr

1 Introduction

The notion of ontology has initially been defined by Gruber as *an explicit specification of a conceptualization* (Gruber, 1993). An ontology is composed of a set of shared classes, properties and relationships with reasoning mechanisms. It has the ability to represent formally real-world knowledge in a shared way. These favorable characteristics of ontologies have led academicians and industrials coming from various communities to adopt them: Artificial Intelligence (Matuszek et al, 2006), Natural Language Processing (Estival et al, 2004), Databases/Data Warehouses (Sugumaran and Storey, 2006; Noy, 2004), Information Retrieval (Graupmann et al, 2005), etc. Some researchers have even pushed the idea of having a universal ontology[1].

The construction of ontologies is time consuming. As a consequence, several research efforts have been conducted to tackle this problem. These studies may be classified into three main categories: (i) *manual construction* as is the case of the Cyc Ontology (Matuszek et al, 2006), (ii) *community-based construction* such as

[1] Entity Relationship Conference 2011 Panel on New Directions for Conceptual Modeling.

7.3.2 The Ontology Definition (ODL), Manipulation (OML) and Query (OQL) Languages of OntoQL

Ontologies and the used ontology language can be managed with the ODL, OML and OQL of OntoQL. These languages have a syntax close to the ones of the DDL, DML and DQL. For example, the next statement, adds the `AllValuesFrom` constructor from OWL to the used ontology language of OntoQL.

```
CREATE ENTITY #OWLRestrictionAllValuesFrom UNDER #Class (
    #onProperty REF(#Property),
    #allValuesFrom REF(#Class) )
```

This statement adds the `OWLRestrictionAllValuesFrom` entity to the core ontology language as a subentity of `Class`. This entity is created with two attributes `onProperty` and `allValuesFrom`. `onProperty` points to a property (`REF(#Property)`) and `allValuesFrom` points to a class (`REF(#Class)`).

An `OWLAllValuesFrom` class can then be created with the OML:

```
INSERT INTO #OWLRestrictionAllValuesFrom
        (#name[en], #onProperty, #allValuesFrom)
  VALUES ('Row_Ball_Bearing', 'uses', 'Row_of_Balls')
```

This example creates the *Row_Ball_Bearing* class as being the set of instances that only use *Row_of_Balls*.

Finally the OQL is used to search ontology elements stored in the OBDB. For example, the next query searches the `OWLAllValuesFrom` classes defined on the `uses` property with the class in which the values of this property must be taken.

```
SELECT #name[en], #allValuesFrom.#name[en]
  FROM #OWLRestrictionAllValuesFrom
 WHERE #onProperty.#name[en] = 'uses'
```

7.3.3 Querying both the ontologies and instances in OntoQL

By combining the DQL and OQL of OntoQL, ontologies and instances can be queried simultaneously.

From ontology to instances. Starting from classes retrieved by a query on ontologies (OQL), the instances of these classes can then be filtered (DQL). This type of queries is possible thanks to *dynamic iterators*. To query instances, an iterator `i` on the instances of a class `C` can be introduced using the `C AS i` construct. OntoQL extends this mechanism with iterators on the instances of a class identified at run-time, which are called dynamic iterators. For example, the following query retrieves the instances of all classes whose names start with `Ball`.

```
SELECT i.oid FROM #class AS C, C AS i
 WHERE C.#name[en] like 'Ball%'
```

From instances to Ontology. Starting from instances retrieved by a DQL query, the description of the belonging classes of these instances can be retrieved (OQL). OntoQL proposes the `typeOf` operator to retrieve the *basis class* of an instance i.e., the minorant class for the subsumption relationship of the classes it belongs to. For example, the following query retrieves the English name of the basis class of each `Rolling_Bearing` instances.

```
SELECT typeOf(b).#name[en] FROM Rolling_Bearing AS b
```

7.4 OntoQL Query Processing

As OntoQL is implemented on top of OntoDB, an OntoQL query is translated into SQL. This process follows five main steps.

1. **OntoAlgebra query plan generation.** The OntoQL query is parsed and turned into an expression tree involving operators of its algebra in its nodes.
2. **OntoAlgebra query plan optimization.** We have identified optimization situations to reduce the OntoAlgebra query plan. These optimizations techniques are detailed in (Jean et al, 2006).
3. **OntoAlgebra query plan translation into relational algebra trees.** This translation is achieved by applying a set of rules described below.
4. **Relational algebra trees optimization.** This step consists in using the different algebraic laws that hold for the relational algebra to turn the relational trees into equivalent trees that may be executed more efficiently by the underlying DBMS.
5. **Relational algebra tree translation into SQL.** The optimized relational trees are translated into SQL queries according to the underlying DBMS and executed to get the *OntoQL* query result.

The translation between OntoQL and SQL query plan is an important step. To illustrate the set of rules defined, table 2 presents a translation rule of an *OntoAlgebra* expression to a relational algebra expression. The interested reader can refer to (Jean, 2007) for the definition of the complete set of rules. In these rules, π represents the projection operator of the relational algebra. `C` is a class and $\mathtt{p}_1, \ldots, \mathtt{p}_n$ are the properties defined on this class. Among these properties only $\mathtt{p}_1, \ldots, \mathtt{p}_u$ are used to describe their instances. The datatype of $\mathtt{p}_1$ is a collection of references and the one of $\mathtt{p}_2$ is a single-valued reference. Others properties have primitive datatypes.

Rule 1 computes the direct instances of the class `C` with their values for all the applicable properties

OntoAlgebra	Relational Algebra
$\texttt{OntoProject}\ (\texttt{C}, \texttt{ext}(\texttt{C}), \{(\texttt{p}_1, \texttt{p}_1), \ldots, (\texttt{p}_n, \texttt{p}_n)\}$	$\pi_{\texttt{Pp}_1\texttt{\_rids},\texttt{Pp}_2\texttt{\_rid},\texttt{Pp}_3,\ldots,\texttt{Pp}_u, \texttt{NULL}\rightarrow\texttt{Pp}_{u+1},\ldots,\texttt{NULL}\rightarrow\texttt{Pp}_n}(\texttt{EC})$

Table 2 Example of a translation rule

on this class. The `OntoProject` operator of *OntoAlgebra* is translated into a projection of the corresponding columns (prefixed by `P`) on the corresponding table (prefixed by `E`). The projections of properties that are not used to describe instances are translated into projections of the `NULL` values as defined in the *OntoAlgebra* semantics. The resulting column is renamed (symbol →) according to the OntoDB naming convention so that other operators can reference it as a used property.

If the identifiers of *width*, *mass* and *Ball_Bearing* are respectively `1`, `2` and `3`, an example of the application of the previous rule is:

```
SELECT width, mass, used_in => SELECT P1, P2, NULL
  FROM Ball_Bearing              FROM E3
```

In this section, we have presented the OntoQL query language. This language has three main characteristics that distinguish it from other proposed languages: (1) the OntoQL language is independent of a given ontology language. Indeed, it is based on a core ontology language, which contains the constructors shared by different ontology languages and this core ontology language can be extended by the OntoQL language itself, (2) the OntoQL language exploits the linguistic information that may be associated to a conceptual ontology allowing users to express queries in different natural languages and (3) the OntoQL language is compatible with SQL. In the next section we present different applications of the OntoDB/OntoQL platform. This platform is available as open source softwares at `http://www.lias-lab.fr/forge/projects/ontodb`.

8 Applications

8.1 Standard Ontologies in Engineering

The PLIB ontology language has been used to develop a number of standard ontologies in different fields of the engineering domain such as:

- Electronic Components (IEC 61360-4);
- Process Instruments (IEC 61360-4);
- Mechanical Fasteners (ISO 13584-511);
- Measure Instruments (ISO 13584-501);
- Cutting Tools (ISO 13399);
- Bearings (ISO 23768);
- Technical Product Documentation (ISO/TC 10 NWI);
- Optics and Photonics (ISO 23584).

To show the size and the complexity of these ontologies, we detail some examples.

The *Mechanical Fasteners* ontology (ISO 13584-511) represents fasteners with their properties and domains of values as they are described in the various ISO mechanical fastener standards. These fasteners include bolts, screws, nuts, rivets, pins and washers. This ontology is composed of approximatively 250 classes and 410 properties. The definitions of these classes are given in French and in English. Several man-years were required for its definition.

The *Cutting Tools* ontology (ISO 13399) defines the terms, properties, and definitions for those portions of a cutting tool that remove material from a workpiece. Cutting items include replaceable inserts, brazed tips, and the cutting portions of solid cutting tools. This ontology is composed of approximatively 500 classes and 360 properties. As for the mechanical fasteners ontology, the definitions of these classes are given in French and in English and several man-years were required for its definition.

The PLIB language has also been used in the eCl@ss classification (`http://www.eclass.de/`). eCl@ss is a product classification and description standard for information exchange between customers and their suppliers. It describes products such as cable, wire, accumulator or battery and services such as process control system or electrical measurement. This classification is composed of approximatively 33 000 classes and described in 15 natural languages.

These example shows the interest of the PLIB ontology language. Next section describes our approach based on the OntoDB/OntoQL platform to integrate heterogenous data.

8.2 Integration a priori and a posteriori

Integrating heterogeneous, autonomous and distributed data sources is one of the favourite application domains of ontologies (Noy, 2004). They contribute on reducing syntactic and semantic heterogeneities that may exist between sources. Due to the increasing number of data sources, automatic integration processes become a necessity for companies. We have proposed data integration solutions for French companies such as Renault S.A. (French multinational vehicle manufacturer established in 1899) which needed to integrate their suppliers data sources. To perform this integration, we claim the following: if we want to avoid human intervention at integration time, mappings between sources shall be done *a priori* during the data sources design. This means that some formal shared ontology must exist, and each data source shall embed some ontological data that reference explicitly this shared ontology. Some integration systems funded by industrials and academic institutions such as the Piscel2 project funded by France Telecom for integrating Web Services (Reynaud and Giraldo, 2003) and the COIN project supported by ARPA and USAF/Rome Laboratory for exchanging financial data (Bressan et al, 2000), worked under the same assumption. Their main limitation is that once the shared ontology is defined, each source shall only use the common vocabulary. The shared ontology is in fact the integrated schema and each source has little schematic autonomy.

To overcome this limitation, we offer more schematic autonomy to data sources participating in the data integration process. To achieve this goal, three assumptions are required.

1. Each data source participating in the integration process shall contain its own ontology. This assumption refers to the need of an ontology-based database (see Section 5).
2. Each data source references a shared ontology *as much as possible*. As much as possible means that: (1) each class of a local ontology references explicitly (or implicitly through its parent class) its lowest subsuming class in the shared ontology, and (2) only properties that do not exist in the shared ontology may be defined in a local ontology; otherwise it should be imported through the *case-of* relationship. This requirement is called *smallest subsuming class reference requirement*.
3. Local ontologies may extend the shared ontology as needed by adding new concepts and properties.

Based on these assumptions, three integration scenarios have been proposed for Renault S.A called: *FragmentOnto*, *ExtendOnto* and *ProjOnto*. These scenarios are described as follows.

FragmentOnto: in this scenario, we assume that the shared ontology is complete enough to cover the needs of all data sources. This scenario is feasible for authoritative parties that can force their suppliers to use the shared ontology. The source autonomy consists in (1) selecting the relevant subset of the shared ontology (classes and properties), and (2) designing the local database schema. This approach has been presented in (Bellatreche et al, 2004). It is not well suited for integrating autonomous data sources.

ExtendOnto: in this scenario, the shared ontology is extended by each local specialization. The local instances are then integrated within the integrated system without any change.

ProjOnto: in this scenario, the shared ontology is not modified. Each data source instance is projected onto the applicable properties of its smallest subsuming class in the shared ontology, and is then added to the population of this class in the shared ontology. These scenarios are well described in the PhD thesis of Dung Nguyen Xuan (Nguyen-Xuan, 2006) and in (Bellatreche et al, 2004).

8.3 Engineering model annotation: the e-Wok Hub Project

The goal of the e-Wok Hub project was to help geologists in carrying out petroleum prospection projects. The petroleum exploration activities is based on the representation of underground reservoirs. These representations use complex and well founded geological models like stratigraphic, structural, datation, geological models, etc. Such an activity leads to a set of complex and heterogeneous models that use a huge amount of data produced either by measurement campaigns or by the corresponding computation models. By heterogeneity, we mean both heterogeneous models in terms of semantics and heterogeneous interpretations since two geologists can give different conclusions when interpreting the models and their corresponding data. Moreover, when working on such projects, engineers and geologists use a large panoply of resources such as scientific articles, reports of past projects, softwares, data files, etc.

The platform developed in the context of this project integrates these heterogeneous resources in a global architecture where ontologies play a central role for data integration, exchange and querying. In the context of this project, the OntoDB database served as a repository and the OntoQL language was used to query the integrated data.

For example, in the case of the CO2 capture and storage, engineers and geologists rely on various engineering models. They have to deal with several interpretation difficulties due to the heterogeneity of these models. To ease this process, we have proposed to annotate these models with concepts of ontologies (Mastella et al, 2009). As the notions of annotations and engineering models were not available in OntoDB, the OntoQL language was used to introduce these notions. First, elements of the engineering models were created with the `CREATE ENTITY` statement of OntoQL. Then, an association table was defined to annotate the engineering models by a class of an ontology. Once this extension was done, OntoQL was used to query the engineering models departing from the ontology concepts.

Moreover, the design of the corresponding ontologies involved several ontologies issued from different domains (stratigraphy, datation, sturcural, etc.) that needed to be integrated in a single ontology without impacting their use in different areas. For this purpose, an *a posteriori case of* relationship has been used to map the different developed ontologies.

This section has shown the interest of the development presented in this paper on several examples of industrial projects. Other successful applications of the PLIB language with its associated technologies include projects with companies such as Toshiba Corporation, Philips, Siemens, Zeiss or Sandvick.

9 Conclusion

In this paper, we have highlighted the benefits of ontologies in the engineering domain. A deep analysis of the use of ontologies by research communities has brought new insights to the classification of ontologies in three main categories: *canonical ontologies*, *non-canonical ontologies* and *linguistic ontologies*. The canonical ontologies described by formalisms such as PLIB have been largely used in the engineering domain. Compared to Semantic Web languages such as RDFS or OWL, PLIB adheres to the closed-world and unique name assumptions, which are adapted to the controlled environment that we found in industrial engineering settings. This language also proceeds to represent the modeling context of the concepts defined in the ontology. This characteristic is important for data integration as an implicit modeling context is the main cause of semantic data heterogeneity (Bressan et al, 2000). This modeling context is expressed by the representation of:

- the context of each class by its set of properties and of each property by its domain of definition;
- the different point of views of the same classes (*view-of operator)*;
- the local interpretation of an ontology by the *case-of* importation mechanism;
- the context of evaluation of a property when it depends upon different parameters;
- the unit and scaling of values.

The PLIB formalism has been largely used to define several standard ontologies in different fields of the engineering domain. The increasing quantity of semantic instances pushes us to propose a persistent solution for them: the OntoDB architecture. The originality of this architecture compared to those defined for RDFS and OWL is twofold: (i) OntoDB leverages the strong typing assumption made in PLIB to propose an efficient storage layout for engineering data and (ii) OntoDB includes the *metaschema* part which has been extensively used to extend the used ontology language (e.g., to store and annotate engineering models with ontologies in the e-Wok Hub project).

The OntoDB database is equipped with the OntoQL language that can be used to define, manipulate and query ontologies and their instances. Instead of defining a language specifically for PLIB, OntoQL is based on a core ontology language composed of the main constructors of several ontology languages. And, this core ontology language can be extended with the OntoQL language itself. Another particularity of this language is that it is compatible with the SQL language: a statement without a namespace definition is considered as a SQL statement. Moreover, the syntax and semantics of OntoQL to query ontologies and their instances are close to the SQL language. Finally, OntoQL exploits the different layers of an ontology, and in particular benefits from the linguistic layer allowing users to write queries in different natural languages. The PLIB language and the OntoDB/OntoQL platform have been used in several industrial projects with companies such as Renault, the French Institute of Petroleum, Toshiba and Philips in various projects related to the problem of data integration.

The developments of the OntoDB/OntoQL platform[5] are continually evolving to satisfy the needs of our partners and of the addressed research issues. These evolutions concern several facets of our platform: (i) *database extension* such as the support of multiple ontology languages, engineering models, semantic web services and functional dependencies, (ii) *query language extension* with user preferences and behaviour semantics, (iii) *database optimization* with the selection of optimization structures such as materialized views and indexes, (iv) *personalization* with query recommendation and relaxation techniques.

Currently, we are also working on probabilistic and fuzzy models for representing uncertainly in ontologies as done in (Huang et al, 2014).

Acknowledgements This work has been initiated under the supervision of the late Professor Guy PIERRA. He has been the initiator of the OntoDB project. We are grateful for his work and his investment. We learned a lot from his experience and advices. The authors would also like to thank the persons who have contributed to the development of the OntoDB/OntoQL platform and in particular Hondjack Dehainsala for the development of OntoDB during his PhD thesis and Dung Xuan Nguyen for his PhD thesis on data integration.

[5] `http://www.lias-lab.fr/forge/projects/ontodb`

References

Abadi DJ, Marcus A, Madden SR, Hollenbach K (2007) Scalable Semantic Web Data Management Using Vertical Partitioning. In: Proceedings of the 33rd International Conference on Very Large Data Bases (VLDB'07), pp 411–422

Aluç G, Özsu T, Daudjee K (2014) Workload Matters: Why RDF Databases Need a New Design. PVLDB 7(10):837–840

Apweiler R, Bairoch A, Wu CH, Barker WC, Boeckmann B, Ferro S, Gasteiger E, Huang H, Lopez R, Magrane M, Martin MJ, Natale DA, ODonovan C, Redaschi N, Yeh LS (2004) Uniprot: the Universal Protein knowledgebase. Nucleic Acids Research 32:D115–D119

Bailey J, Bry F, Furche T, Schaffert S (2005) Web and Semantic Web Query Languages: A Survey. In: Reasoning Web, pp 35–133

Bechhofer S, van Harmelen F, Hendler J, Horrocks I, McGuinness DL, Patel-Schneider PF, Stein LA (2004) OWL Web Ontology Language Reference. World Wide Web Consortium, URL `http://www.w3.org/TR/owl-ref`

Bellatreche L, Pierra G, Xuan DN, Hondjack D, Ait-Ameur Y (2004) An a Priori Approach for Automatic Integration of Heterogeneous and Autonomous Databases. In: Proceedings of the 15th International Conference on Database and Expert Systems Applications (DEXA'04), pp 475–485

Bressan S, Goh CH, Levina N, Madnick SE, Shah A, Siegel M (2000) Context Knowledge Representation and Reasoning in the Context Interchange System. Applied Intelligence 13(2):165–180

Brickley D, Guha R (2004) RDF Vocabulary Description Language 1.0: RDF Schema. World Wide Web Consortium, URL `http://www.w3.org/TR/rdf-schema/`

Broekstra J, Kampman A, van Harmelen F (2002) Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In: Proceedings of the First International Semantic Web Conference (ISWC'02), pp 54–68

de Bruijn J, Lara R, Polleres A, Fensel D (2005) OWL DL vs. OWL Flight: Conceptual Modeling and Reasoning for the Semantic Web. In: Proceedings of the 14th International Conference on World Wide Web (WWW'05), pp 623–632

Cullot N, Parent C, Spaccapietra S, Vangenot C (2003) Ontologies : A contribution to the DL/DB debate. In: Proceedings of the first International Workshop on Semantic Web and Database (SWDB'03), pp 109–129

Das S, Chong EI, Eadon G, Srinivasan J (2004) Supporting Ontology-Based Semantic matching in RDBMS. In: Proceedings of the 13th International Conference on Very Large Data Bases (VLDB'04), pp 1054–1065

Dehainsala H, Pierra G, Bellatreche L (2006) Managing Instance Data in Ontology-based Databases. Tech. rep., LISI/ENSMA, URL `http://www.lisi.ensma.fr/ftp/pub/documents/reports/2006/2006-LISI-003-DEHAINSALA.pdf`

Dehainsala H, Pierra G, Bellatreche L (2007) OntoDB: An Ontology-Based Database for Data Intensive Applications. In: Proceedings of the 12th International Conference on Database Systems for Advanced Applications (DASFAA'07), pp 497–508

Duan S, Kementsietsidis A, Srinivas K, Udrea O (2011) Apples and oranges: a comparison of RDF benchmarks and real RDF datasets. In: Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'11), pp 145–156

Estival D, Nowak C, Zschorn A (2004) Towards Ontology-Based Natural Language Processing. In: Proceedings of the 4th Workshop on NLP and XML (NLPXML'04), pp 59–66

Graupmann J, Schenkel R, Weikum G (2005) The Sphere-Search Engine for Unified Ranked Retrieval of Heterogeneous XML and Web Documents. In: Proceedings of the 31st International Conference on Very Large Data Bases (VLDB'05), pp 529–540

Gruber TR (1993) A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition 5(2):199–220

Harris S, Gibbins N (2003) 3store: Efficient Bulk RDF Storage. In: Proceedings of the 1st International Workshop on Practical and Scalable Semantic Systems (PSSS'03), pp 1–15

Huang HD, Lee CS, Wang MH, Kao HY (2014) IT2FS-based ontology with soft-computing mechanism for malware behavior analysis. Soft Computing 18(2):267–284

IEC61360-4 (1999) Standard data element types with associated classification scheme for electric components - Part 4 : IEC reference collection of standard data element types, component classes and terms. Tech. rep., International Standards Organization

Jean S (2007) OntoQL, un langage d'exploitation des bases de données à base ontologique. PhD thesis, LISI/ENSMA and University of Poitiers

Jean S, Aït-Ameur Y, Pierra G (2006) Querying Ontology Based Database Using OntoQL (an Ontology Query Language). In: Proceedings of On the Move to Meaningful Internet Systems 2006: CoopIS, DOA, GADA, and ODBASE, OTM Confederated International Conferences (ODBASE'06), pp 704–721

Jean S, Aït-Ameur Y, Pierra G (2007a) An Object-Oriented Based Algebra for Ontologies and their Instances. In: Proceedings of the 11th East European Conference in Advances in Databases and Information Systems (ADBIS'07), pp 141–156

Jean S, Pierra G, Ameur YA (2007b) Domain Ontologies: A Database-Oriented Analysis. In: Web Information Systems and Technologies, International Conferences, WEBIST 2005 and WEBIST 2006. Revised Selected Papers, pp 238–254

Knorr M, Alferes JJ, Hitzler P (2011) Local Closed World Reasoning with Description Logics under the Well-Founded Semantics. Artificial Intelligence 175(9-10):1528–1554

Ma L, Su Z, Pan Y, Zhang L, Liu T (2004) RStar: an RDF Storage and Query System for Enterprise Resource Management. In: Proceedings of the 30th International Conference on Information and Knowledge Management (CIKM'04), pp 484–491

Maio CD, Fenza G, Furno D, Loia V, Senatore S (2012) OWL-FC: an upper ontology for semantic modeling of Fuzzy Control. Soft Computing 16(7):1153–1164

Mastella LS, Aït-Ameur Y, Jean S, Perrin M, Rainaud JF (2009) Semantic exploitation of persistent metadata in engineering models: application to geological models. In: Proceedings of the IEEE International Conference on Research Challenges in Information Science (RCIS 2009), pp 147–156

Matuszek C, Cabral J, Witbrock M, Deoliveira J (2006) An Introduction to the Syntax and Content of Cyc. In: Proceedings of the 2006 AAAI Spring Symposium on Formalizing and Compiling Background Knowledge and Its Applications to Knowledge Representation and Question Answering, pp 44–49

Mendes PN, Jakob M, Bizer C (2012) DBpedia: A Multilingual Cross-domain Knowledge Base. In: Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC'12), pp 1813–1817

Neumann T, Weikum G (2008) RDF-3X: a RISC-style engine for RDF. PVLDB 1(1):647–659

Nguyen-Xuan D (2006) Intégration de base de données hétérogènes par articulation a priori d'ontologies : application aux catalogues de composants industriels. PhD thesis, LISI/ENSMA and University of Poitiers

Niles I, Pease A (2001) Towards a Standard Upper Ontology. In: Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001), pp 2–9

Noy NF (2004) Semantic Integration: A Survey Of Ontology-Based Approaches. SIGMOD Record 33(4):65–70

OMG (2002) Meta Object Facility (MOF), Specification v1.4, OMG Document formal/02-04-03

Paasiala P, Aaltonen A, Riitahuhta A (1993) Automatic Component Selection. In: Proceedings of the 9th CIM-Europe annual conference on Realising CIM's industrial potential, pp 303–312

Pan Z, Heflin J (2003) DLDB: Extending Relational Databases to Support Semantic Web Queries. In: Proceedings of the 1st International Workshop on Practical and Scalable Semantic Systems (PSSS'03), pp 109–113

Park MJ, Lee JH, Lee CH, Lin J, Serres O, Chung CW (2007) An Efficient and Scalable Management of Ontology. In: Proceedings of the 12th International Conference on Database Systems for Advanced Applications (DASFAA'07), pp 975–980

Pierra G (2003) Context-Explication in Conceptual Ontologies: The PLIB Approach. In: Proceedings of the 10th ISPE International Conference on Concurrent Engineering (CE 2003), pp 243–254

Prud'hommeaux E, Seaborne A (2008) SPARQL Query Language for RDF. W3C Recommendation 15 January 2008 http://www.w3.org/TR/rdf-sparql-query/

Reynaud C, Giraldo G (2003) An Application of the Mediator Approach to Services over the Web. In: Special track Data Integration in Engineering, Concurrent Engineering (CE'2003), pp 209–216

Sakr S, Al-Naymat G (2009) Relational Processing of RDF Queries: A Survey. SIGMOD Record 38(4):23–28

Schenk D, Wilson P (1994) Information Modelling The EXPRESS Way. Oxford University Press

Schmidt M, Hornung T, Lausen G, Pinkel C (2009) SP^{2}Bench: A SPARQL Performance Benchmark. In: Proceedings of the 25th International Conference on Data Engineering (ICDE'09), pp 222–233

Sequeda J, Tirmizi SH, Corcho Ó, Miranker DP (2011) Survey of Directly Mapping SQL Databases to the Semantic Web. Knowledge Engineering Review 26(4):445–486

Shaw GM, Zdonik SB (1990) A Query Algebra for Object-Oriented Databases. In: Proceedings of the 6th International Conference on Data Engineering, pp 154–162

Sidirourgos L, Goncalves R, Kersten ML, Nes N, Manegold S (2008) Column-Store Support for RDF Data Management: not all swans are white. PVLDB 1(2):1553–1563

Suchanek FM, Kasneci G, Weikum G (2008) YAGO: A Large Ontology from Wikipedia and WordNet. Journal Web Semantics 6(3):203–217

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

Weiss C, Karras P, Bernstein A (2008) Hexastore: Sextuple Indexing for Semantic Web Data Management. PVLDB 1(1):1008–1019

Wilkinson K (2006) Jena Property Table Implementation. In: Proceedings of the 2nd International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS'06), pp 35–46

Yuan P, Liu P, Wu B, Jin H, Zhang W, Liu L (2013) TripleBit: a Fast and Compact System for Large Scale RDF Data. PVLDB 6(7):517–528

Zou L, Mo J, Chen L, Özsu MT, Zhao D (2011) gStore: Answering SPARQL Queries via Subgraph Matching. PVLDB 4(8):482–493