Domain Ontologies: A Database-Oriented Analysis

Stéphane Jean, Guy Pierra, and Yamine Ait-Ameur

Laboratory of Applied Computer Science (LISI)
National Engineering School for Mechanics and Aerotechnics (ENSMA) - Poitiers
86960 Futuroscope Cedex, France
{jean,pierra,yamine}@ensma.fr

Abstract. If the word ontology is more and more used in a number of domain, the capabilities and benefits of ontology for Information Systems management are still unclear. Therefore, the usage of ontology-based Information Systems in industry and services is not widespread. This paper analyses the concept of a domain ontology from a database perspective. As a result, firstly, we provide three criteria that distinguish domain ontology from other existing domain modeling approach which lead us to propose a new definition of domain ontologies. Secondly, based on the various approaches of ontology modeling followed by different communities, we propose a taxonomy of domain ontology. We show how they may be organized into a layered model, called the onion model, allowing to design and to use the capabilities of each category of ontology in an integrated environment. Finally, this paper presents several information systems based on ontology technologies and describe the kinds of services that should be provided to allow a powerful usage of ontology in data management.

Keywords: Ontology, Semantic Web, Ontology Based Information Systems, Semantic Integration, Data Exchange, PLIB.

1 Introduction

Defined by T. Gruber (Gruber, 1993) as an explicit specification of a conceptualization, an ontology may be considered as a quite new and exciting artefact in computer science allowing to represent explicitly meaning. Nowadays, the word ontology is used in a lot of diverse research fields including natural language processing, information retrieval, electronic commerce, Web Semantic, software component specification and information systems integration. In this context, several proposals for ontology models and languages and corresponding operational systems have been developed in the last decade. The growth of both the number and the diversity of such models for ontologies leads to some difficulties encountered by engineers when they need to identify the right ontology(ies) and the right ontology model(s) to use or to apply in practical engineering areas.

Due to the wide domain of usage, the meaning of the word ontology is of course context-dependent. Borrowed from philosophy, where it stand for "a systematic account of existence"[1], the term ontology got a quite new meaning in technical and

[1] FOLDOC: http://foldoc.org

J. Filipe, J. Cordeiro, and V. Pedrosa (Eds.): WEBIST 2005/2006, LNBIP 1, pp. 238–254, 2007.

computer science fields. In this new context, one may distinguish *upper-level* or *foundation ontologies*, the goal of which is to provide definition for general-purpose concepts, such that process, object or event, and to act as foundation for more specific domain ontologies (Niles and Pease, 2001; Gangemi et al., 2003), and *domain ontologies* that are tied to a specific universe of discourse and model the corresponding domain knowledge.

The goal of this paper is to analyse the concept of a domain ontology in a database perspective. Most of the usual definitions, such that the one of the Free On-line Dictionary of Computing "an explicit formal specification of how to represent the objects, concepts and other entities that are assumed to exist in some area of interest and the relationships that hold among them" are so broad that they covers most of the previous information modelling artefacts such that conceptual models, knowledge model or specification of information exchange formats. As a result the high potential of ontologies for semantic integration is hidden, and a number of engineers consider ontology as a buzzword.

In this paper, we suggest that the above definition should be refined and we propose three criteria to distinguish domain ontologies from other information modelling artefacts. Such domain ontologies introduce a new modelling level in the database field and we propose a taxonomy of the various possible domain ontologies together with integration scenarios that show how this taxonomy may be helpful for addressing various data management issues. Then we discuss what kind of tools, called Ontology-based Data Management System (OBDMS) would be useful for promoting ontology usage in the data processing community. It is expected that this analysis will promote the development of new OBDMS and help engineers and practitioners to choose relevant OBDMS in order to solve their business problems. Our work is different from previous related work (Cullot et al., 2003; Meersman, 2001) aiming at clarifying the differences between ontology and database technologies. The main contribution of this paper are the following:

- a proposal for criteria that distinguish domain ontologies from other domain modeling approaches;
- a new definition of domain ontology;
- a taxonomy of domain ontology and a layered model (the onion model) that shows how these different kinds of ontology may cooperate for solving data processing issues.

This paper is organized as follows. Next section presents the concept of an ontology by focussing on three criteria that distinguish ontology from other existing modeling approach. This suggest a new definition of domain ontologies. Section 3 describes the various possible usages of ontologies in data management. A database-oriented taxonomy of ontologies is proposed in section 4 and section 5 proposes an integrated view of these different kinds of domain ontology for addressing various data processing problems. Finally, section 6 describes several OBDMS and compares their capabilities.

2 Specificity of Domain Ontology as Domain Models

We propose in this section three criteria that characterize domain ontologies. These criteria suggest a new definition of domain ontology. Finally, we discuss the difference between domain ontologies and conceptual models.

2.1 Ontology Criteria

From our point of view, a domain ontology is a domain conceptualization obeying to the three following criteria.

1. **Formal.** An ontology is a conceptualization based on a formal theory which allows to check some level of consistency and to perform some level of automatic reasoning over the ontology-defined concepts and individuals. We note that this criterion excludes most meta-models that do not provide automatic reasoning capabilities.
2. **Consensual.** An ontology is a conceptualization agreed upon by a community larger than the members involved in one particular application development. For instance, The Gene Ontology (GO) project[2] is a collaborative effort between more than 10 organisms to address the need for consistent descriptions of gene products. Moreover, users are invited to submit suggestions for improving the GO ontologies. ISO 13584-compliant (PLIB) product ontologies follow a formal standardization process and are published as ISO or IEC international standards. We note that this criterion excludes most database, conceptual models which are just tailored for a particular database application.
3. **Capability to be referenced.** Each ontology-defined concept is associated with an identifier allowing to refer to this concept from any environment, independently of the particular ontology model where this concept was defined. We note that this criterion exclude, in particular, all specification of information exchange formats, such that STEP (Standard for the Exchange of Product Model Data) Application Protocols (ISO10303, 1994), where entities and attributes may only be referenced from the specified exchange structure.

2.2 A Proposed Definition for Domain Ontology

These three criteria lead us to propose a new definition for domain ontology. For us, a domain ontology is a *formal and consensual dictionary of categories and properties of entities of a domain and the relationships that hold among them*. By entity we mean being, i.e, anything that can be said to be in the domain. The term dictionary emphasizes that any entity of the ontology and any kind of domain relationship described in the domain ontology may be referenced directly, for any purpose and from any context, independently of other entities or relationships, by a symbol. This identification symbol may be either a language independent identifier, or a language-specific set of words. But, whatever be the symbol, and unlike in linguistic dictionary, this symbol denotes directly a domain entity or relationship, the *description* of which is formally *stated* providing for automatic reasoning and consistency checking.

[2] http://www.geneontology.org/

We show in the next section that the criteria used for characterizing domain ontologies allow to distinguish them with previous kind of concept modeling like conceptual models and knowledge models.

2.3 Ontologies vs. Conceptual Models

As both an ontology and a conceptual model define a conceptualization of a part of the world, an ontology seems similar to a conceptual model. Conceptual models respect the *formal* criterion. Indeed, a conceptual model is based on a rigorously formalized logical theory and reasoning is provided by view mechanisms. However, a conceptual model is application requirement driven: it *prescribes* and *imposes* which information will be represented in a particular application (logical model). Two different application systems having always at least slightly different application requirements, conceptual models are always different from systems to systems. Thus, conceptual models do not fulfill the *consensual* criterion. Moreover, an identifier of a conceptual model defined concept is a name that can only be referenced unambiguously inside the context of an information system based on this particular conceptual model. Thus, conceptual models also do not fulfill the *capability to be referenced* criterion.

In the same manner, a conceptualization defined in Knowledge Representation and Artificial Intelligence using logic constructors are not, in general, an ontology. Such conceptualizations satisfy the *formal* criterion. Indeed, logic is equipped with formal semantics that enables automatic reasoning. However, in such knowledge models, the main goals are the inference capabilities of the models. Before that the notion of an ontology emerged, neither mechanisms for referencing each particular concept of a knowledge model, nor processes for ensuring a consensus on the concepts were considered. Therefore, like conceptual models, such usual knowledge models do not fulfill the *consensual* and the *capability to be referenced* criterion.

However, data model constructs issued from database design and logic are suitable for ontology models definitions. Indeed, several ontology models, like OWL (Bechhofer et al., 2004), RDFS (Brickley and Guha, 2004) and KAON (Bozsak et al., 2002) for description logic and PLIB (Pierra, 2003), DOGMA (Jarrar and Meersman, 2002) and MADS (Parent et al., 1999) for database design, are based on constructors provided either by database conceptual models or by artificial intelligence knowledge base models. These models add other constructors that enable to satisfy the *consensual* criterion (context definition, multi instantiation, separation between concept definition and data structure prescription) and *the capability to be referenced* criterion (URI, GUI).

On the basis of this distinction between ontology and other concepts models, we study in next section what ontologies are good for.

3 What Are Ontologies Good For?

As stated in the introduction, ontology technologies is widespread in a lot of diverse application domains and it may be used in various engineering steps like specification, data exchange, data integration and search.

3.1 Specification

Two usages of ontologies as specification are reported.

The usage of a conceptualization as a specification is the basis of the Model-Driven Architecture (MDA). A model of the application is first defined. This model is then used to generate the code of the application. The existing formal link between the specification and the software enables to evolve the code when the specification evolves. Currently, several softwares addressing similar problems on the same domain are defined using different conceptualizations. This makes difficult interoperation between these softwares. Ontology usage is a solution to this problem. Because ontologies satisfy the *consensual* criterion, the various conceptualization corresponding to various domain softwares may be connected to a domain ontology. Then, softwares can interoperate using accessors provided by this ontology. This approach is called ontology-driven software engineering (Tetlow et al., 2005).

The same approach can be followed in database design. The proposition of Ontology-Based Database approach (Pierra et al., 2005) is to use an ontology as a first level of database concepts specification. This ontology is then specialized to define a conceptual model. Because all particular systems have particular requirements, different conceptual models may be built on the same consensual ontology. The link between ontologies, conceptual and logical models is kept inside a database. This architecture enables the evolution of both the conceptual model and of the ontology and provides a common access to information through the ontology. The advantage of this approach is to make clear what is common between two systems, and what is different.

3.2 Data Exchange

A consensual domain conceptualization that can be referenced may easily be used as an interchange format for data over this domain (ISO13584-42, 1998; Chawathe et al., 1994). Unlike usual exchange format that specify the complete structure of the exchanged data and where the meaning of each piece of data results from its place in the *global* structure, ontology-based exchange are very flexible. In such an exchange, the meaning of each piece of data may be defined *locally*, by referencing ontology identifiers. This allow quite different exchange structures to be soundly interpreted by the same receiving system.

3.3 Data Integration

Domain ontologies is the only artefact that allows to reconcile, at the semantics level, heterogenous data source models. When domain ontologies are explicitly represented in databases, the integration may be fully automated even when each source specializes locally the shared ontology (Bellatreche et al., 2004). In the Semantic Web approaches, the link between source and ontology is usually supported by metadata. The integration is often automatic because the ontologies used in this process capture and identify concepts in a formal and unique way.

In natural language processing, the link between sources and an ontology consists of the words contained in the documents. Most of the words having context-dependent meaning, the integration process is often user-assisted to provide meaningful results.

3.4 Data Access and Search

An ontology provides an access to data that reference the concepts it defines. Depending on the expressive power of the ontology model, data may be browsed using the is-a concept hierarchy, queried using keyword or with more sophisticated query languages.

Ontologies are also used to query databases. The approach consists in enriching the queries on the logical model by expressions involving ontology-defined concepts and expressions (Das et al., 2004).

To sum up, ontology applications are widespread. For a complete survey describing various usages of ontology, the interested reader can refer to (Uschold and Jasper, 1999). However all ontologies are not similar. Next section proposes a taxonomy of ontologies to highlight their differences and the consequences on their usage for the various application domains seen previously.

4 A Taxonomy of Domain Ontologies

Several orthogonal criteria need to be used to classify ontologies. The first major criterion is the manner of conceptualizing a domain. Indeed, a domain can be conceptualized as a set of words or as a set of concepts. This conceptualization way leads to the distinction between linguistic (taxonomic) ontologies and conceptual (descriptive) ontologies (Cullot et al., 2003; Pierra, 2003). Following (Pierra, 2003), we call Linguistic Ontologies (LO) those ontologies whose scope is the representation of the meaning of the words used in a particular Universe of Discourse, in a particular language. On the other hand, Conceptual Ontologies (CO) are those whose goal is the representation of the categories of objects and of the properties of objects available in some part of the world.

As these two kinds of ontology address quite different problems and fields, it is fundamental to clarify which kind of ontology is suited in each particular business context. Before presenting our taxonomy, let us review some fundamentals of ontologies.

4.1 Fundamentals of Ontologies

Concepts defined in a conceptual ontology can be classified in two categories.

- *Primitive concepts* are those concepts "for which we are not able to give a complete axiomatic definition" (Gruber, 1993). Here, the definition relies on a textual documentation and a knowledge background shared between the readers. The set of primitive concepts define the border of the domain conceptualized by an ontology. Primitive concepts are the ground on which all other ontology concepts will be built. The definition of primitive concepts being always, at least partially, informal, the only quality criteria, one has for such definitions, is that they represent a consensus over some community. Without such a consensus one cannot asset the usability of an ontology.
- Besides primitive concepts, a number of ontology models focus on the capability to create conservative definitions (Gruber, 1993), i.e, to associate a new term or a new concept to something that is already defined by another mean in the ontology under design. This characteristic is the basis of inference mechanisms like automatic

classification. *Defined concepts* are those concepts for which the ontology provides a complete axiomatic definition by means of necessary and sufficient conditions expressed in terms of other concepts (either primitive concepts or other defined concepts).

When defined concepts are introduced, *concept equivalence* relation needs to be defined in order to be able to compare, classify or relate defined and/or primitive concepts.

Concept equivalence can be defined at the class level. This is the approach followed by models based on Description Logics (DL) like OWL (Bechhofer et al., 2004) or on the Carin language used in the PICSEL project (Rousset et al., 2002). For example, in PICSEL the concept of Hotel is defined as a specialization of the primitive concept HousingPlace. A HousingPlace is defined as a place having associated buildings, rooms and meal services. A hotel is then fully defined as those Housing Places which have more than five rooms and have only CollectiveBuilding.

Concept equivalence can also be expressed at the property level. This is the case in models where derivation functions can be defined. F-Logic (Kifer et al., 1995) is one of the models supporting this capability. For example, the property ”boss” relating an employee to another employee can be derived from the properties ”belong to” and ”chair”.

Thanks to the distinction between primitive concepts and defined concepts, we are now able to propose our database-oriented taxonomy of ontologies.

4.2 Canonical Conceptual Ontology (CCO)

In database design, when conceptual model is defined, ambiguity or possible multi-representation of the same fact is forbidden. The same approach is followed for performing exchanges between two different databases. It consists in defining a canonical vocabulary in which each information in the target domain is captured in an unique way without defining any synonymous constructs. For example, in the STEP project, exchange models are defined in the EXPRESS language as a STEP Application Protocol (AP). These canonic exchange models are used by industrial users to exchange product descriptions between different organizations.

Ontologies whose definitions follow this approach are called *Canonical Conceptual Ontologies*. In CCOs, each domain concept is described in a single way, using a single description that may include necessary condition. As a consequence, CCOs include *primitive concepts* only.

Defining CCOs is the main goal of the ontology model defined in the PLIB (Part Library) standard (ISO13584-42, 1998) with a first focus on representing and exchanging formal ontologies of technical domains. Such CCOs use a property-based characterization of the involved concepts. This means that a class is only created when it proves necessary for defining the domain of some properties. Therefore, in PLIB, the generalization/specialization concept hierarchies are rather ”flat”. An example of such an ontology for electronic components can be found in (IEC61360-4, 1999).

These examples show that usage of CCOs in the context of data exchange is fruitful. More arguments are given latter in this paper by studying an exchange scenario.

4.3 Non Canonical Conceptual Ontology (NCCO)

In database design, concept equivalence plays an important (but second-order) role. Each database addresses a particular domain and defines a canonic way of representing any fact about this domain. Then, in order to achieve a degree of independence with respect to the choice of the concepts offered to users, a database designer provides *views*. These *defined concepts* are specified using the CREATE VIEW operator on the basis of the primitive concepts that constitute the database schema. In deductive database this functionality also exists, provided with derived entities.

So, whatever the construct offered to express concept equivalence, we call *Non Canonical Conceptual Ontologies* those ontologies which contain not only primitives concepts but also defined concepts.

NCCOs are particulary useful when they are used as global query schemas. For example, in the PICSEL project mentioned earlier, primitive concepts of local CCOs are expressed as defined concepts for the primitive concepts of a global ontology used as global query schema. In the global ontology, concept expressions and rules expressed on basic concepts of the tourism domain (*HousingPlace*, *Flight* ...) define a wide range of terms (*Hotel*, *Bed&Breakfast* ...) useful for users to formulate a query on data referenced by local CCOs.

NCCO constructors are also very useful to define mappings between different ontologies. For example, figure 1 presents two CCOs of the domain of wines. The CCO (A) is property oriented while the CCO (B) is entity oriented.

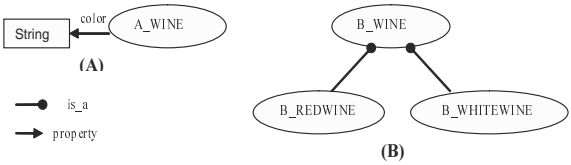

Fig. 1. Local Wine CCO

Applying some NCCO operators, issued from a DL syntax, we can write that:

$B\_REDWINE \equiv A\_WINE \sqcap color : red$

This axiom states that the concept of B_REDWINE defined in CCO (B) is equivalent to the concept A_WINE defined in CCO (A) restricted by the value of the color attribute. These primitives have formal semantics which enables a powerful reasoning mechanism implementation in tools named reasoners (e.g, RACER (Haarslev and Möller, 2001)). These tools support mechanisms for concepts and instances classification. As a result, the two local CCOs can be automatically merged into the NCCO of figure 2. This NCCO provides a global access to data of the two CCOs.

In this example, the reasoner has inferred that all instances of $B\_REDWINE$ are instances of A_WINE having the value *red* for the property *color*. When these facts are materialized, as it is proposed in OntoMerge (Dou et al., 2003), it become possible to split the merged ontology into the CCO (A) and the CCO (B) with the new inferred instances. This process shows that NCCO constructors are useful for integration tasks.

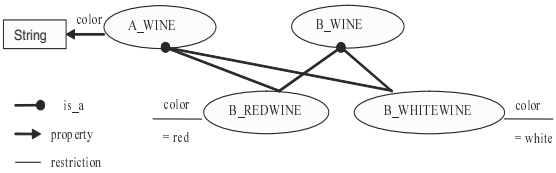

Fig. 2. Integrated Wine NCCO

4.4 Linguistic Ontologies

In other application domains like information retrieval or natural language processing, human languages play a key role. Even in database, natural languages are used in various places. Indeed, table and attribute names are chosen to reflect their meaning. Moreover, conceptual model documentation is largely, and in a number of cases completely, expressed in natural language.

We call Linguistic Ontology (LO) those ontologies that define words or contextual usages of words. In this kind of ontology only words relationships (synonym, hyponym, ...) are available. Words relationships being highly contextual, machine inference in general needs expert supervision. Moreover, approximate relationships may be defined.

Wordnet (http://www.cogsci.princeton.edu/wn) is the most well-known representative of this category of ontologies. It provides with textual definitions, synonymous terms and representations for the various concepts that can be denoted by a term. They are intended to be used as a sophisticated thesauri.

LOs help to recognize conceptual similarities between sentences even if different terms are used. Since word meanings are contextual and their relationships are approximative, wrong similarities may be produced and results can never be considered as reliable. An example of LOs usage is given in (Das et al., 2004). A LO on types of cooking is used to retrieve more meaningful results on the food served in a restaurant. Thus, querying the 'latin american' served food retrieves tuples having 'american' or 'mexican' as value of type of cooking. A number of semi-automatic database integration approaches, e.g., (Visser et al., 1999), use such linguistic ontologies.

4.5 Discipline Specific View of Domain Ontology

Currently, the three categories of ontology mainly correspond to three different disciplines of computer science and have few connexions. Ontology models focus either on CCO, NCCO or LO design.

CCO are mainly considered in the data processing community. In CCOs, ontology descriptions focus on primitives concepts characterization and identification.

- They include precise and complex descriptions of the primitive concepts. These descriptions are provided using CCO oriented model constructs. For example, the DOGMA formalism (Jarrar and Meersman, 2002), a database inspired ontology model, provides contextual identification of concepts. In the PLIB model, primitive concepts can be associated with reference to real documents, pictures, usage restrictions .... This model also distinguishes the rigid properties

(Guarino and Welty, 2002), i.e., properties essential for any instance of a class from those that may or not hold or exist according to the role in which an entity is involved.
- They don't contain model mappings. Consequently, the encoding of such conversions is achieved in an application.

NCCO models were developed by artificial intelligence community. Therefore, they focus on inference and concept equivalence.

- NCCOs contains conservative definition of defined concepts using useful operators like boolean operators (union of classes ...).
- As a rule, they include less precise descriptions for the primitive concepts. For example, in OWL, a primitive concept description is limited to a label, a comment and the properties (roles) that can apply to its instances.

LOs were designed for computational linguistic. In LOs (e.g Wordnet), each word is associated with several synsets (sets of synonymous) that reflects its various meanings. The imprecision of the conceptualization is due to the following facts:

- words meaning depends upon a context;
- words relationships (e.g synonymy) have no formal definition whereas concepts relationships (e.g subsumption) have.

5 Relationship Between Ontology Categories and Proposal for a Layered Model

The previous observations lead us to identify some relationships between CCOs, NCCOs and LOs.

- Mappings between CCO might also be defined as equivalence operators of some NCCO;
- NCCO models can use powerful CCO-oriented model constructs to define their own primitive concepts;
- LOs might define the various meaning of each word of a particular language by reference to a NCCO. This reference would provide a basis for formal and exact reasoning and automatic translation of context-specific terms.

As a further step towards this observation, we first propose a layered model for domain ontology design. Then, we present an example of usage of this layered model.

5.1 A Layered Model for Ontology Design

An often used guide (Noy and McGuinness, 2001) proposes a seven steps approach for NCCO development.

1. Determine the domain and scope of the ontology to be developed.
2. Consider reusing existing available ontologies that someone else has developed.

3. List the important terms in the ontology without considering the possible concepts overlaps they may lead to.
4. Define the classes and the associated class hierarchy. From the list created in step 3, select the terms that describe objects having independent existence. These terms will be classes in the ontology and will become anchors in the class hierarchy.
5. Define the properties associated to classes. Indeed, most of the remaining terms are likely to be properties of these classes.
6. Define the constraints (cardinality, domain and range restrictions) that hold on properties.
7. Create instances of classes in the hierarchy.

This approach exploits the NCCO capability to define equivalent concepts and thus to integrate several ontologies addressing the same domain. Since we claim that NCCOs can benefit from being articulated with CCOs, we propose an alternative approach for the development of a NCCO starting from a CCO.

1. The first step of the design of an ontology should be to agree within a community on a CCO. To reach this agreement, it is required to:
   - define clearly what is the domain covered by the ontology;
   - choose a powerful model to define precisely the primitive concepts existing in the domain and allow to explicate the evaluation context of a property value (Pierra, 2003);
   - provide a common understanding of a canonic set of concepts covering the domain. This conceptualization must accommodate a wide and diverse range of technical and business requirements shared by the members of the community. It must reach a wide recognition and acceptance.
2. Within the community of users and/or developers, on the basis of the defined CCO, a NCCO may be built for use by members of this community either to build their own view of the domain or to model formally all the concepts existing in the target domain that are associated with a usual linguistic denotation (word or sequence of words). Proceeding this way ensures the preservation of the capability to share and exchange information expressed in terms of the CCO.
3. In order to allow the use of the defined NCCO for linguistic inference and/or to provide an end-user friendly user interface in various languages, a list of language-specific terms needs to be defined and associated to each concept in the NCCO.

According to this alternative approach, figure 3 illustrates a layered model, called the onion model, of the resulting domain ontology. A CCO provides a formal basis to model and to exchange efficiently the knowledge of a domain. A NCCO provides mechanisms to relate different conceptualizations made on this domain. Finally, LO provides natural language representation of the concepts of this domain, possibly in the various languages where these concepts are meaningful.

5.2 An Exchange Scenario Based on Layered Domain Ontology

In database universe, each database uses a canonical vocabulary. Usually, each database uses a different canonical vocabulary. Instead of defining a NCCO covering all the

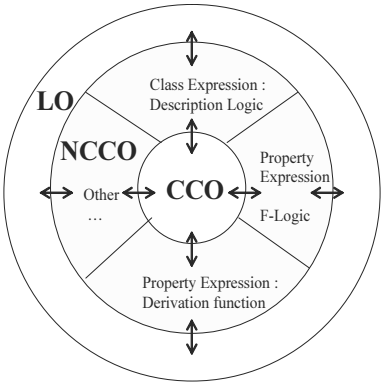

Fig. 3. The onion model of domain ontology

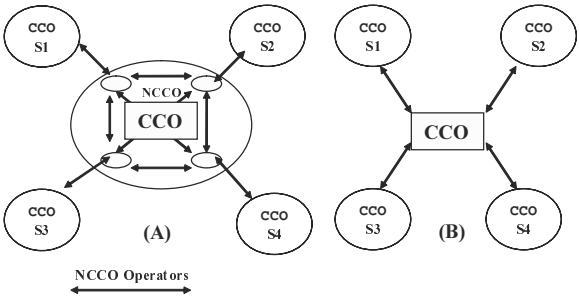

Fig. 4. Use of ontologies for canonic data exchange

terms of all the sources (see fig 4 (A)), the onion model suggests that all exchanges are performed using a consensual CCO. Each source just contains the descriptions of its own concepts in terms of the (primitive) concepts of the canonical conceptual ontology (see fig 4 (B)). This approach has been put into practice for databases integration in (Bellatreche et al., 2004). Notice that if each concept is represented differently in the n participating sources, and if there exists in each source a mean value of p concepts, solution (A) needs to implement $n * p$ mappings in each source, when solution (B) requires only p mappings in each source. Moreover, this approach also applies to virtual exchange, i.e, mediator (Wiederhold, 1991).

This clause, and the above show the interest of articulating the three categories of ontologies according to our onion model across the whole life cycle of domain ontologies.

- CCOs provide canonical and accurate descriptions of each concept of a given domain. It provides sound basis for exchange between different sources.
- NCCO operators are used to interact with other applications or sources that have already their own particular ontologies.
- LOs provide linguistic capabilities over primitive and defined concepts.

Next section discusses the various tools that would be useful to facilitate the use of ontologies in data management activity and compare with those currently existing.

6 Ontology Based Data Management System (OBDMS)

An OBDMS is a suite of tools providing support for using ontologies and ontology instances data. In data management, many different functionalities may be expected from an OBDMS. In this section we list a set of such functionalities and relate them to our taxonomy.

6.1 OBDMS Functionalities

An OBDMS may provide the following functionalities.

- **F1. Respect of standard.** If a formal semantics is defined for some supported ontology model standard, this semantics shall be respected by the defined OBDMS.
- **F2. Handling of exchange format.** When an exchange format is provided with an ontology model, import/export ontologies and ontologies instance data in this format are needed.
- **F3. Data manipulation.** An OBDMS should provide support to insert, update or delete ontology concepts and instance data (support of CCO).
- **F4. Linguistic support.** An OBDMS should support and exploit linguistic-oriented naming and description of concepts in various languages (support of LO).
- **F5. Data querying.** An efficient way should be available for querying both ontologies and ontologies data.
- **F6. Ontology Mapping.** An OBDMS should provide mapping functionalities to integrate different ontologies (support of NCCO operators).
- **F7. Ontology as a specification.** An OBDMS should provide the possibility to extract from an ontology a specification of a software or of a database schema.

6.2 Comparison of Some OBDMS Implementations

The description of various useful capabilities needed to facilitate the use of ontologies in data management being completed, this section reviews three OBDMS according to the functionalities they offer.

6.2.1 RDF Tools

RDF Suite and Sesame are two suites of tools for RDFS storage and querying. A database is used for the persistence of the data. These two tools add constraints on the RDFS standard. They propose import/export module to manage data described in an RDF syntax. Third-party tools can be associated to this suite by using this module. Thus, an ontology editor (e.g. Protege) can be associated to these tools. However, this editor will not be synchronized with the storage system. Querying data in Sesame and

RDF Suite is performed through the RQL language. There is no support provided by these tools to integrate different ontologies or to use them as a specification.

6.2.2 PLIB Suite

PLIB Suite (http://www.plib.ensma.fr) is a set of tools for PLIB ontologies storage, edition, querying and integration. The exchange format of PLIB ontologies is EXPRESS physical file. PLIB Suite include full support of this format. Currently, ontologies storage rely on the OntoDB prototype. This prototype is an implementation of the notion of Ontology Based Database presented in section 3. Thus, it stores ontologies, data and logical models of data defined by extraction and/or specialization from ontologies. It respects full definition of the PLIB standard and its architecture is flexible enough to manage the evolution of this standard. This prototype is directly linked to the PLIB Editor software that can be used to visualize and manage ontologies and data. Each concept is associated to a name in different natural languages with synonymous names. PLIB Editor exploits this natural language capability to provide a multilingual interface. However, there is no linguistic inference using synonymous names. PLIB Editor provides a query module that enables to build visually a query on data from the ontologies. This module relies on the OntoQL query language allowing to manage and to query both ontologies and data. Integration of PLIB ontologies is enabled when subsumption links are defined between different ontologies.

6.2.3 RacerPro

RacerPro is an in-memory OWL reasoning system. RacerPro supports OWL DL almost completely and includes a graphical user interface to manipulate ontologies and data. These data are directly persisted in an OWL file. Querying these data is possible through the query language nRQL. However, memory demands, concurrency control and scalability of this implementation are still open issues. As a payoff, RacerPro offers a full reasoning system that enables automatic classification of concepts and of data. As we have seen in section 4.3, this functionality is very useful for integration.

Table 1 summarizes the previous study.

Table 1. Fulfilled functionalities by existing OBDMS

	RDF Tools	Racer Pro	PLIB Suite
F1	partial	yes	yes
F2	yes	yes	yes
F3	partial	yes	CCO
F4	no	no	partial
F5	yes	partial	yes
F6	no	NCCO	partial
F7	no	no	yes

It appears that none of the existing tools covers the complete set of functionalities required to implement the scenario proposed in section 5.2. Therefore, next section proposes an architecture using existing OBDMS to solve this problem.

6.3 An Integrated Architecture to Implement Our Data Exchange Scenario

Our exchange scenario requires one consensual CCO managed in an OBDMS providing the possibility to map NCCOs (F6). Since Racer is the only existing OBDMS providing this functionality we propose to use it for this purpose.

Each source defines its own CCO and manages its instances. We propose to use PLIB Suite for each particular source. The use of PLIB Suite for this purpose ensure that a precise description of the concepts will be available (F3). Moreover it provides a scalable repository to store all the data (F5). For linguistic support (F4), it provides the necessary resources to reference a LO such as Wordnet.

The exchange of data requires the following steps:

1. A source exports its primitive concepts to Racer (F2). They become the consensual CCO. Notice that this step requires that PLIB Suite can export data in OWL format.
2. Other sources may use Racer editor to define mapping between their concepts and the defined consensual CCO.
3. Using the Racer reasoning system, a *classification of all the concepts* is done. Thus, the consensual CCO become a NCCO.
4. Data instances to be exchanged between the defined sources are imported into Racer.
5. Using the Racer reasoning system, a *classification of these instances* is performed.
6. For concepts of a given source under which instances have been classified, an export of these instances to this source is achieved. This step requires to keep track of the origin of each concept.

Other propositions like (Pan and Heflin, 2003) are made to associate a reasoner with a database. However the aims of these propositions is to provide a full reasoning system using a database. Yet, OWL reasoning is not fully supported by these systems.

7 Conclusion

This paper has investigated the concept of domain ontology in a database perspective. Ontology becoming a buzzword, often used as a new term for already existing models, we have first proposed three criteria to characterize ontology. A domain ontology must be formal, i.e, allowing some automatic reasoning and consistency checking capability, consensual in some community and able to be referenced from any environment. These three criteria characterize domain ontology as a new kind of model in computer science and lead us to propose a new definition of domain ontology as a formal and consensual dictionary of categories and properties of entities of a domain and the relationships that hold among them.

Domain ontology models being mainly developed by three communities, we have proposed a taxonomy of domain ontology into CCO, NCCO and LO. After reviewing the partial domain coverage of these various models, we have proposed a layered model, called the onion model of domain ontology, allowing to design and to use the capabilities of each category of ontology in an integrated environment. We have also discussed under the name of OBDMS what kinds of services should be provided to allow a powerful usage of ontology in data management.

Currently there exist neither exchange format nor OBDMS able to represent and to manage domain ontologies corresponding to the complete onion model. First, we are developing an XML schema that will integrate both OWL and PLIB capabilities. Second, we are extending the PLIB Suite to support OWL class expression constructs using both a connexion with an OWL reasoner and a representation by SQL views. Finally, we are working on a query language, called OntoQL, allowing to query the three layers of the onion model.

Acknowledgements

The authors would like to thank the anonymous referees of this paper. Their relevant comments and suggestions were very helpful to improve the quality of this paper.

References

Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D.L., Patel-Schneider, P.F., Stein, L.A.: OWL Web Ontology Language Reference. World Wide Web Consortium (2004)

Bellatreche, L., Pierra, G., Xuan, D.N., Hondjack, D., Ait-Ameur, Y.: An a priori approach for automatic integration of heterogeneous and autonomous databases. In: Galindo, F., Takizawa, M., Traunmüller, R. (eds.) DEXA 2004. LNCS, vol. 3180, pp. 475–485. Springer, Heidelberg (2004)

Bozsak, E., Ehrig, M., Handschuh, S., Hotho, A., Maedche, A., Motik, B., Oberle, D., Schmitz, C., Staab, S., Stojanovic, L., Stojanovic, N., Studer, R., Stumme, G., Sure, Y., Tane, J., Volz, R., Zacharias, V.: Kaon - towards a large scale semantic web. In: Bauknecht, K., Tjoa, A.M., Quirchmayr, G. (eds.) EC-Web 2002. LNCS, pp. 304–313. Springer, Heidelberg (2002)

Brickley, D., Guha, R.: RDF Vocabulary Description Language 1.0: RDF Schema. World Wide Web Consortium (2004)

Chawathe, S.S., Garcia-Molina, H., Hammer, J., Ireland, K., Papakonstantinou, Y., Ullman, J.D., Widom, J.: The tsimmis project: Integration of heterogeneous information sources. In: IPSJ, pp. 7–18 (1994)

Cullot, N., Parent, C., Spaccapietra, S., Vangenot, C.: Ontologies: A contribution to the dl/db debate. In: Bussler, C.J., Tannen, V., Fundulaki, I. (eds.) SWDB 2004. LNCS, vol. 3372, pp. 109–129. Springer, Heidelberg (2005)

Das, S., Chong, E.I., Eadon, G., Srinivasan, J.: Supporting ontology-based semantic matching in rdbms. In: Aberer, K., Koubarakis, M., Kalogeraki, V. (eds.) Databases, Information Systems, and Peer-to-Peer Computing. LNCS, vol. 2944, pp. 1054–1065. Springer, Heidelberg (2004)

Dou, D., McDermott, D., Qi, P.: Ontology translation on the semantic web. In: Proceeding of the 2nd International Conference on Ontologies, Databases and Applications of Semantics (ODBASE'2003), pp. 952–969 (2003)

Gangemi, A., Guarino, N., Masolo, C., Oltramari, A.: Sweetening wordnet with dolce. AI Magazine 24(3), 13–24 (2003)

Gruber, T.R.: A translation approach to portable ontology specifications. Knowl. Acquis. 5(2), 199–220 (1993)

Guarino, N., Welty, C.: Evaluating ontological decisions with ontoclean. Commun. ACM 45(2), 61–65 (2002)

Haarslev, V., Möller, R.: Racer system description. In: Goré, R.P., Leitsch, A., Nipkow, T. (eds.) IJCAR 2001. LNCS (LNAI), vol. 2083, pp. 701–706. Springer, Heidelberg (2001)

IEC61360-4. 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. Technical report, International Standards Organization (1999)

ISO10303. Initial release of international standard(is) 10303. Technical report is 10303, International Standards Organization (1994)

ISO13584-42. Industrial automation systems and integration parts library part 42 : Description methodology : Methodology for structuring parts families. Technical report, International Standards Organization (1998)

Jarrar, M., Meersman, R.: Formal ontology engineering in the dogma approach. In: Meersman, R., Tari, Z., et al. (eds.) CoopIS 2002, DOA 2002, and ODBASE 2002. LNCS, vol. 2519, pp. 1238–1254. Springer, Heidelberg (2002)

Kifer, M., Lausen, G., Wu, J.: Logical foundations of object-oriented and frame-based languages. J. ACM 42(4), 741–843 (1995)

Meersman, R.: Ontologies and databases: More than a fleeting resemblance. In: OES/SEO Workshop Rome (2001)

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

Noy, N.F., McGuinness, D.L.: Ontology development 101: A guide to creating your first ontology. Technical report ksl-01-05 and stanford medical informatics technical report smi-2001-0880, Stanford Knowledge Systems Laboratory (2001)

Pan, Z., Heflin, J.: Dldb: Extending relational databases to support semantic web queries. In: PSSS (2003)

Parent, C., Spaccapietra, S., Zimanyi, E.: Spatio-temporal conceptual models: data structures + space + time. In: GIS '99: Proceedings of the 7th ACM international symposium on Advances in geographic information systems, pp. 26–33. ACM Press, New York (1999)

Pierra, G.: Context-explication in conceptual ontologies: The plib approach. In: Jardim-Goncalves, R., Cha, J., Steiger-Garcao, A. (eds.) Proceedings of the 10th ISPE International Conference on Concurrent Engineering (CE 2003), pp. 243–254 (2003)

Pierra, G., Dehainsala, H., Aït-Ameur, Y., Bellatreche, L.: Base de données à base ontologique: principes et mise en œuvre. Ingénierie des Systèmes d'Information 10(2), 91–115 (2005)

Rousset, M.-C., Bidault, A., Froidevaux, C., Gagliardi, H., Goasdou, F., Reynaud, C., Safar, B.: Construction de médiateurs pour intégrer des sources d'information multiples et hétérogènes: Picsel. revue I3 2(1), 9–59 (2002)

Tetlow, P., Pan, J., Oberle, D., Wallace, E., Uschold, M., Kendall, E.: Ontology Driven Architectures and Potential Uses of the Semantic Web in Systems and Software Engineering. World Wide Web Consortium (2005)

Uschold, M., Jasper, R.: A framework for understanding and classifying ontology applications. In: Proceedings of the IJCAI99 Workshop on Ontologies and Problem-Solving Methods(KRR5), Stockholm, Sweden (August 1999)

Visser, P.R.S., Beer, M.D., Bench-Capon, T.J.M., Diaz, B.M., Shave, M.J.R.: Resolving ontological heterogeneity in the kraft project. In: Bench-Capon, T.J.M., Soda, G., Tjoa, A.M. (eds.) DEXA 1999. LNCS, vol. 1677, pp. 668–677. Springer, Heidelberg (1999)

Wiederhold, G.: Obtaining information from heterogeneous systems. In: Proceedings of the First Workshop on Information Technologies and Systems (WITS'91), pp. 1–8. Cambridge MA, MIT Sloan School of Management (1991)