International Journal of
Soft Computing and Software Engineering (JSCSE)

Vol.5, No.3, 2015
Published online: March 25, 2015

e-ISSN: 2251-7545
DOI: 10.7321/jscse.v5.n3.1

Designing a Data Warehouse from OWL sources

[1]Yassine Laadidi, [2*]Mohamed Bahaj
[1,2]*Departement of Mathematics and Computer Science, Faculty of Science and Technology, University Hassan 1st, Settat, Morocco*
Email: [1]yassine.laadidi@gmail.com, [2]mohamedbahaj@gmail.com

Abstract. **The Semantic web is the new extension of actual web to make data "understandable" by computers in order to construct one global source of information. As a result, a huge quantity of semantic data is provided. RDF (which stands for Resource Description Framework) is a standard model for data, used and designed to describe information on the semantic web. Web Ontology Language (OWL) is the standard language used to describe semantic relationships and allows us to specify far more about the properties and classes. A data warehouse as a dimensional schema is designed to change and grow up over time to respond to the business needs. This paper describes our approach to define a dimensional fact model from OWL ontology sources. The method treats a complex ontology structure in two parties, first by a simplification process that allows us to clean up and focus in important concepts and needed data, the second party is the construction of the dimensional fact model according to the resulting OWL structure from the previous party.**

Keywords: *RDF, OLAP, OWL-DL, BI, Semantic Web, Ontology, Star-Schema*

* Corresponding Author:
Mohamed BAHAJ,
Faculty of Science and Technology,
University Hassan 1st, Settat, Morocco.
Email: mohamedbahaj@gmail.com Tel:+212660250506

1. Introduction

Business Intelligence (BI) can be defined as the process of turning data into information and then into knowledge [1]; therefore to store and historicize that information data warehouses have been developed.

Data warehouse is implemented through an ETL process which is in charge of extracting data from different sources, cleaning, transforming and finally delivering data into the data warehouse. On-line analytical processing (OLAP) is a process that allows us to view data through a data warehouse from different axes and at several levels of aggregations. A view is also called an OLAP cube which is a multidimensional database working under control of an OLAP server (for performance reasons), however, with the huge volume of data provided in a consistent way and by different sources, that process could be very limited without taking into consideration factors like availability of data sources, level of details and auto-updating.

The semantic web is a huge source of information which "things", represented as nodes, are related to each other. In fact, if we navigate through those nodes that compose the semantic web we can

e-ISSN: 2251-7545
DOI: 10.7321/jscse.v5.n3.1

realize that it expresses facts represented as triples. The semantic web grows up with every contribution of web users (auto-updating) which provide a high level of details about resources, and thus makes the information much more important and valuable.

Recommended by W3C [2], the Web Ontology Language (OWL) is the standard language used to create ontologies, it extends RDF and RDFS, its primary aim is to define and instantiate ontologies.

To benefit of semantic data sources and extract useful and relevant information, this paper presents an approach that is based on user specifications, simplifying a complex OWL ontology structure to a another one more specific to users business needs, and as a final step generating a dimensional fact model.

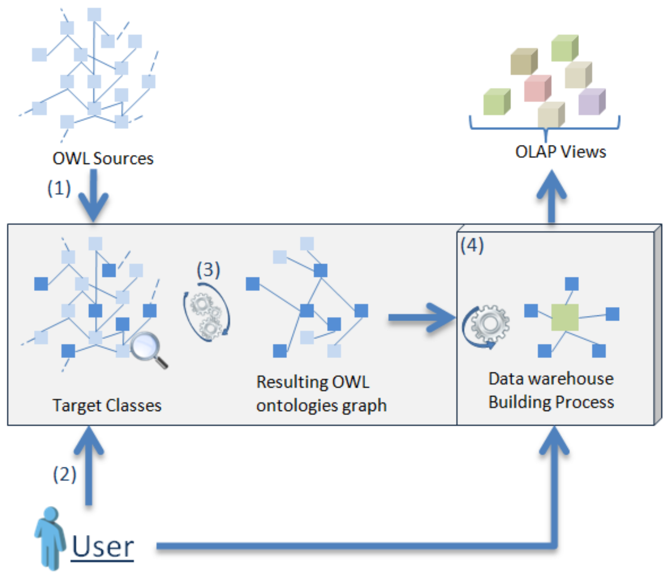

Figure 1. Simplification and transformation of OWL structure

Figure 1 represents the method adopted in this paper to simplifying and transform OWL sources into a dimensional fact model. The system work as follows:

1. Extracting and building the global OWL structure (i.e. the set of triples that compose the ontology) using OWL ontology sources.
2. The user of the system chooses and identifies the target concepts (i.e. classes in OWL language) based on his business needs.
3. By taking into consideration the target classes, the next process is clearing the OWL structure by eliminating unnecessary classes and simplifying it as much as possible.
4. Defining the dimensional fact model from the resulting OWL structure.

This paper is organized as follows: in section 2, we briefly review related works. In section 3, we present the OWL-DL language and dimensional data modeling. In section 4 we explained the method of simplification of OWL structure. In section 5 we present how the system can define a dimensional model on resulting OWL structure. Finally, a conclusion is given in section 6.

2. Related work

Many approaches have been developed to manipulate semantic data and OWL ontology sources in purpose to build data warehouses. The approach in [6] presents a method for identification and

International Journal of
Soft Computing and Software Engineering (JSCSE)

Vol.5, No.3, 2015

e-ISSN: 2251-7545
DOI: 10.7321/jscse.v5.n3.1

extracting valid facts, in this method the analyst pick up concepts and properties the ontology to construct multidimensional star schema.

The approach in [5] adopts another point of view in dealing with ontology sources. The designer or analyst of the data warehouse must choose an object property as the fact, from this point on a dependency graph is created and afterword transformed into a star schema.

However, [5] and [6] are limited and treat only specific OWL ontology sources which are already oriented to a specific business-domain and limited to OWL-Lite version, thus, they cannot deal with a large complex OWL-DL ontology. Other approaches [3 and 4] have also been developed to define the passage from OWL ontology sources to a relational database but do not include dimensional modeling.
In this paper, the approach presented is characterized by two primary phases: the first phase consists of simplifying a complex ontology graph based on designer business-needs and constructs a simple ontology and the second phase to make series of transformations and define a multidimensional star schema resulting ontology source.

3. Overview

In OWL language, a concept is represented as a class and roles, represented as properties. A class represents a set of individuals sharing some characteristics, for example: *owl:thing* is the root class which represents the set of all individuals, also class hierarchies can be stated by using *rdfs:subClassOf* property, for example: if x is in instance of class C and class C is subclass of C', then x is instance of C' as well.

OWL defines two types of properties: data type properties and object properties. A data type property represents a relationship between a resource and a data type value. An object property specifies a relationship between two resources. In its second version, OWL ontology language offers many modeling features and present advanced class relationships and an advance use of properties and data types.

A class expression can be formed by applying boolean combinations (i.e. using the standard operators union, intersection and complement) to other named classes, also a class expression can be represented as an enumerated class by using *owl:oneOf* property and *rdf:parseType="Collection"* to enumerate individuals which are sharing some characteristics or by placing restrictions on object and data property expressions and on cardinality of object and data property. A class can also be defined in terms of a class or classes that is disjoined with and which have no instances in common. The property *owl:disjointWith* is used in OWL language to indicate disjoined classes.

OWL2 provides other modeling capabilities with properties as the inverse property, symmetric/asymmetric properties, transitivity, reflexivity, etc. Object or data type properties can be gathered as collection that is used as a key for identifying a class expression.

Domain and range restrictions allow us to identify the subject element and the object (or value) in a triple and thus determine the direction of the property in the graph as shown in the example in figure 2.

e-ISSN: 2251-7545
DOI: 10.7321/jscse.v5.n3.1

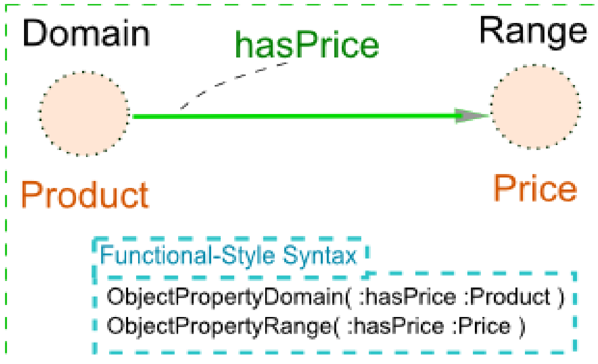

Figure 2. Domain and range restriction example

A data warehouse according to [7] is a subject-oriented, integrated, nonvolatile, and time-variant collection of data. The principle aim of dimensional modeling is to present the data in a standardized form and facilitate interrogation from the conceptual point of view as it is related to two concepts of fact, dimension and hierarchy.

In a relational system, the fact table stores all relevant measures (i.e. numerical property describes a quantitative attribute that is relevant to analysis) that characterize the analyzed subject (e.g. quantities, amounts, volume sales, etc). It also contains substitution keys to all related dimensions tables.

A Dimension table presents a description of an axis, for example, typical dimensions for the sales fact are products, stores, and dates. Sometimes attributes of a dimension can be organized according to their level of detail. To define these various levels, each dimension is provided with one or more hierarchies.

Those concepts which compose the dimensional model can be structured in many dimensional schemas, one of the well-known is the star schema illustrated in figure 3 and adopted in our approach.

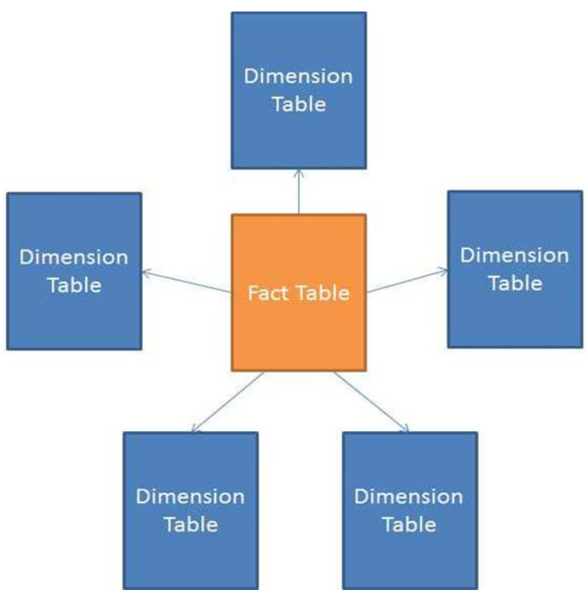

Figure 3. Example of a Star-Schema

e-ISSN: 2251-7545
DOI: 10.7321/jscse.v5.n3.1

4. Identification and Simplification

In this section, we shall focus on the simplification process of the OWL ontology structure as a source of information. This stage is very important because it will allow us to clear up the actual OWL structure and keep only what is considered valued and important for the final process which is the definition of the dimensional fact model.

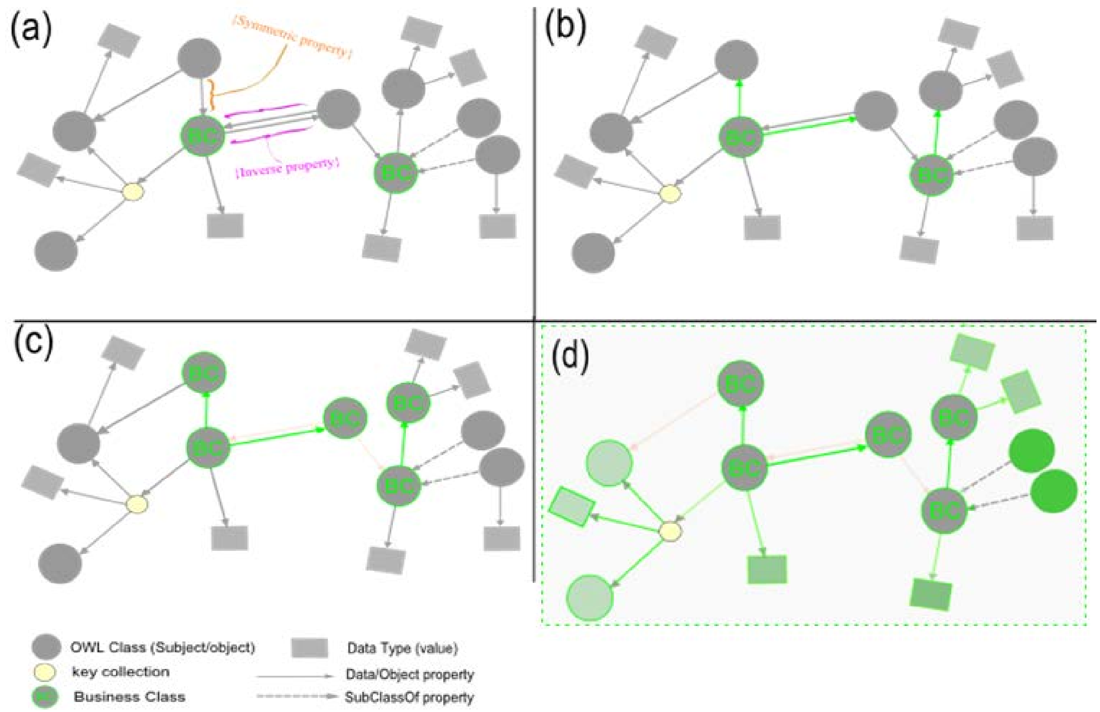

Figure 4. Simplification of an OWL structure

The first task in this process is the identification of business classes. In this section we shall define business classes as classes considered by the user as important for responding to his business needs. The user or designer identifies and selects classes from the OWL ontology structure as an initialization process, however, for more performance and coherence, the designer must choose classes which are not related directly to each other like the example shown in (figure 4 a).

After selecting business classes, the second task is to design a new OWL ontology structure based on the existing one. The idea is simple:

- Step one, the first business classes (see figure 4 a) are selected by the user and put on the list.
- Step two, selecting all object properties which have as domain a class from the business classes list (see figure 4 b).
- Step three, selecting all classes which are the ranges of the previous object properties and add them onto business classes list (figure 4 c).
- Step four, from all classes contained in the business classes list, selecting all data type properties which have as domain a business class (figure 4 d).
- Finally, from previous data properties, selecting all data types (figure 4 d).

International Journal of
Soft Computing and Software Engineering (JSCSE)

Vol.5, No.3, 2015

e-ISSN: 2251-7545
DOI: 10.7321/jscse.v5.n3.1

All objects (i.e. classes and data types) which not selected (illustrated in figure 4 by grey circles) during the execution of the algorithm are not added to the new OWL structure. The result of this phase is shown in (figure 4 d).

Other simplifications, however, are not mentioned which are related to other aspects like class hierarchies, complex classes and properties characteristics that are related to objects in our OWL structure and should be treated in the next phase of our simplification algorithm which complete the first phase (i.e. no business classes must be removed).

Class hierarchies

In OWL ontology language, a class A is *subClassOf* a class B which means that every instance of A is an instance of B, from this definition we assume that subclasses could be seen not only as specialization of a concept but also as a level of detail, however, we may find complex class hierarchies (i.e. a class may have subclasses and these later also have subclasses and so on…) as shown in figure 5, therefore to resolve this problem, we believe that only subclasses of the first level of class hierarchy are required to be included in the OWL structure to represent details about the super class.

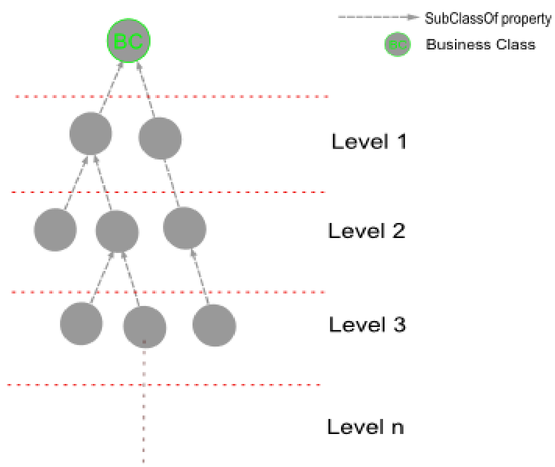

Figure 5. Class hierarchies

Advanced class expressions

In the previous section we have seen how named classes, properties and individuals can be used as building blocks to define new classes and thus express complex knowledge, for example: a customer can be a person or an organization, (i.e. union of two concepts) this sentence is expressed in functional-style syntax as follow:

```
EquivalentClasses(
  : Customer
  ObjectUnionOf(:Person :Organization)
)
```

We assume, however, that class expressions are not part of this process, although they are involved in the ETL process, because all those complex classes will help to populate the data warehouse with coherent and correct data. Consequently, all class expressions are removed from the OWL structure.

e-ISSN: 2251-7545
DOI: 10.7321/jscse.v5.n3.1

Properties characteristics

In its latest version, OWL provides an addition of class expressions, many other modeling capabilities with properties.

An inverse property is specified between two properties in which one is the inverse of the other. This situation is already taking on consideration by our algorithm in step two of the first phase which allows us to select objects only in one direction (illustrated by green arrows in figure 4).

Symmetric property characteristic is also taken into account directly after identification of business classes (i.e. between step one and two in the first phase) as shown in the passage from (a) to (b) in figure 4.

A collection of (object or data) properties can be assigned as a key to a class. At the end of the first phase, all objects or data type which are related to a business class by a key collection are kept to our new OWL structure as shown in the result of our example (figure 4 d).

At this point, our OWL ontology structure is more specific and ready to be used as an OWL source to define our dimensional fact model.

5. Defining the dimensional fact model

In the previous section the designer has made his choices about the primary concepts (i.e. first classes selected by himself) judged needed and required in data warehouse building process, those concepts (or classes in OWL language) are directly transformed into dimensions. However, the designer could also identify more classes as potential dimensions, but a condition must be respected, every chosen class must not be a range of any object property. From this point, the dimensional fact model can be defined.

The process begins by applying some transformations regarding some aspects that we saw before as class hierarchies and key collection, defining dimensions, hierarchies, attributes, and last but not least, finish by constructing the fact. In this section, we will use the resulting OWL structure used in the example in figure 4 to apply our transformations and define the star schema behind it. Transformations are as follows:

Transformation 1:

All subclasses selected during the simplification process (see figure 4 d) are grouped and transformed into a single node (or class) and related to the super class (see figure 5).

Transformation 2:

All objects (classes or data types) that compose the key collection and characterize an actual business class are become directly related to this later on (figure 5).

Transformation 3:

A class not recognized as dimension and has at least one data type property is automatically removed, and therefore all data type properties are related directly to the dimension class (see passage from figure 4 d to figure 5 a).

International Journal of
Soft Computing and Software Engineering (JSCSE)

Vol.5, No.3, 2015

e-ISSN: 2251-7545
DOI: 10.7321/jscse.v5.n3.1

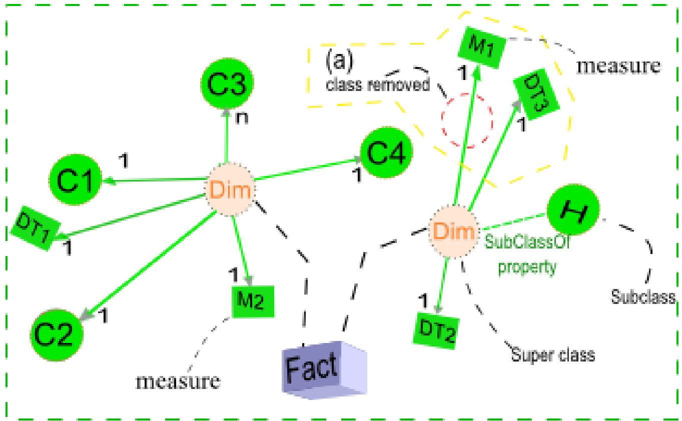

Figure 6. Example of transformations

It is always possible for the designer to make his own modifications by removing useless classes after these transformations and by selecting potential measures from the existing data types.

Dimensions are already identified, the next step is to define hierarchies and attributes. To define a hierarchy or an attribute we will be based in cardinalities, if the object property has cardinality greater than 1 then the range class must be a hierarchy table (for example class C3 in figure 6) and if it has cardinality equal to 1 the range class will be an attribute in dimension table (for example class C1, C2 and C4 in figure 6). The subClassof property remains a special case, after transformation 1, the resulting node (see class H in figure 6) is defined as a hierarchy table. Also, ranges of all data type properties (see DT1, DT2 and DT3 in figure 6), except selected ones (i.e. measures), are transformed into attributes.

At this point, the fact table of the dimensional model will be defined, all data types selected previously as potential measures are transformed to attribute in the fact table and all dimensions are related to this later by surrogate keys. We must insist that every passage from a class in OWL ontology language to a table in SQL Language, a primary key must be created. The resulting star schema for our example is shown in figure 7.

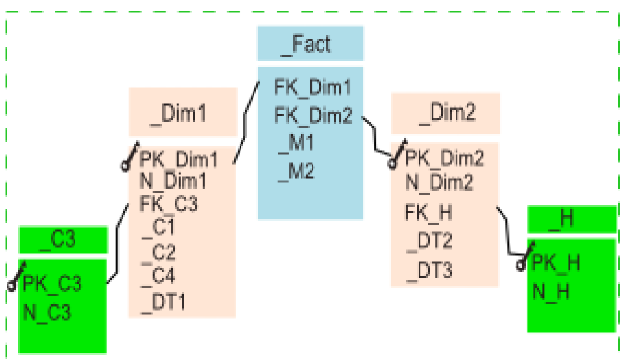

Figure 7. The resulting star schema for the example

6. Conclusion

In this paper, we presented our approach to define dimensional fact model from OWL ontology sources. The method treats an OWL ontology structure in two parties, first by a simplification process that allows us to clear up and focus on important concepts and needed data, the second party is

e-ISSN: 2251-7545
DOI: 10.7321/jscse.v5.n3.1

the construction of the dimensional fact model according to the resulting OWL structure from the previous party. This approach allows the data warehouse designer more flexibility while using complex OWL ontology sources.

References

[1] M.Golfarelli, S.Rizzi, and Iuris Cella, Beyond Data Warehousing: What's Next in Business Intelligence , 7th ACM international workshop on Data warehousing and OLAP, New York, USA, 2004.
[2] World Wide Web Consortium, W3C Semantic Web Activity, Web source www.w3.org/, 2014.
[3] X. Liu, DataWarehousing Technologies for Large-scale and Right-time Data, dissertation, Faculty of Engineering and Science at Aalborg University, Denmark, 2012.
[4] E. Vysniauskas, and L. Nemuraite, Mapping of OWL ontology concepts to RDB schemas, Proceedings of the 15th International Conference on Information and Software Technologies, pp. 317– 327, Lithuania 2009.
[5] M. Gulić, Transformation of OWL Ontology Sources into Data Warehouse, Faculty of Maritime Studies, Rijeka, Croatia, 2013.
[6] V.Nebot, R.Berlanga, Building DataWarehouses with Semantic Data, Lenguajes y Sistemas Informáticos, Castellón, Spain, 2010.
[7] W.H. Inmon, Building the Data Warehouse, 3rd edition, John Wiley, New York, 2002.
[8] D.Allemang, J.Hendler, Semantic Web for the Working Ontologist, Effective Modeling in RDFS and OWL, second Edition, May 2011.
[9] T.Heath, C.Bizer, Linked Data Evolving the Web into a Global Data Space, Morgan & Claypool Publishers, 2011.
[10] B.DuCharme, Learning SPARQL Querying and Updating with SPARQL 1.1, second edition, August 2013.
[11] D. Skoutas, A. Simitsis, and T. K. Sellis, "Ontology-Driven Conceptual Design of ETL Processes Using Graph Transformations," J. Data Semantics, vol. 13, pp. 120–146, 2009
[12] M. Niinim¨ aki and T. Niemi, "An ETL process for OLAP using RDF/OWL ontologies," J. on Data Semantics XIII, vol. 5530, pp.97–119, 2009.
[13] O. Romero, A. Simitsis, and A. Abell ´ o, "GEM: Requirementdriven generation of ETL and multidimensional conceptual designs," in 13th Int. Conf. in Data Warehousing and Knowledge Discovery (DaWaK), ser. Lecture Notes in Computer Science, vol. 6862. Springer, 2011, pp. 80-95
[14] Susan F Ellakwa, El-sayed El-azhary and Passent El-kafrawy.,(Integrated Ontology for Agricultural Domain. International Journal of Computer Applications 54(2):46-53, September 2012