Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online
CONFERENCE PAPER ARCHIVE

IFAC-PapersOnLine 48-4 (2015) 220–225

Ontology versus Database

Michal Sir ** Zdenek Bradac **
Petr Fiedler

** Department of Control and Instrumentation, Brno University of Technology Czech Republic (Tel:+420541141308; e-mail: sir@feec.vutbr.cz).*
*** Department of Control and Instrumentation, Brno University of Technology Czech Republic (email: bradac@feec.vutbr.cz).*
**** Department of Control and Instrumentation, Brno University of Technology Czech Republic (email: fiedlerp@stud.feec.vutbr.cz)*

Abstract: The goal of this paper is to clarify the differences between ontologies and databases. The article describes, in a step-by-step manner, the parts in which differences occur. However, there are also similarities proving that ontologies and databases are not completely different. Based on these aspects, this paper presents various approaches to transforming a database to ontology. The conclusion summarizes and highlights the most important similarities and differences.

Keywords: Ontology, database, structure, differences, similarity

1 INTRODUCTION

Some of us still remember the time when the computers were as large as a ballroom. During the period, however, most data (such as library booklists, borrowing records or lists of employers) were safe in file cabinets, which allowed data organization according to different criteria to classify a new entry. Most file cabinet operations were performed by people; data elimination was nevertheless carried out using paper shredder machines, a tool already existing at that time.

Currently, computers dominate the world, and therefore the above-discussed data types are stored in computer warehouses. Being replaced with databases, file cabinets are progressively disappearing; in recent years, databases and ontologies have been widely applied. While databases are already well known and used as a part of the everyday working routine, ontologies are gaining in popularity only gradually (despite the increasing preferences). Both of these "organized data warehouses" are, however, becoming integral elements of our lives.

1.1 Historical differences

From the historical point of view, these two domains are very different. If we intend to compare ontology and database before they became IT instruments, we have to return hundreds of years into the past. At the time of its first use, ontology was a philosophical discipline; Aristoteles worked with the concept and referred to it as "the first philosophy" (384 – 322 BC), (Sir et al. 2011). He defined ontology as the doctrine of being. Another major period in the development of the notion was the Renaissance, during which functionalist ontology was created by Nicolaus Cusanus. In modern philosophy, the concept came to be associated with rationalism, an approach that represents the union of being and thinking.

As mentioned above, the history of databases began somewhat differently; although we cannot exactly determine when file cabinets were created, this probably occurred at the time the first library attempted to register the first volume. The next stage of databases consisted in the actual transfer of filing cabinets to machines; this period could be dated to the 1890s, a time when a large census was organized in the USA. Generally, then, data were stored on punched cards, and the processing was carried out on electromechanical machines. A higher-level computer language, COBOL, appeared in the 1960s. This language was then used for many years as a significant data processing tool. In 1971, the Database Task Group (DBTG) issued a report titled *The DBTG April 1971 Report*. The report included proposals of a database scheme, a language for the scheme definition, and a subscheme. Another part of this document was a complete description of a networked database system. Around 1970, relational database was mentioned for the first time, and its inception is ascribed to E. F. Codd. The modern history of databases then began in the 1990s; it was in this decade that object-oriented databases emerged. Although these databases were originally to replace the old relational model, the attempt failed, and object-relational database was created as a compromise between relational and object-oriented databases.

From the evolutional point of view, ontology underwent more prominent development than database: it started as a philosophical discipline analyzing existence and being and gradually expanded into information technologies. Databases (or, previously, file cabinets) have always served the same purpose: today, they are used similarly as in the past. The major difference in this context consists in the changed agent or operator, namely in the move from humans to computers.

10.1016/j.ifacol.2015.07.036

2 BASIC DIFFERENCES

For a thorough explanation of the characteristics of database and ontology, several key terms have to be defined. The first such concept is the UNA (Unique Name Assumption); the definition of this term asserts that there is only one word available for one entity from the real world. (Poote and Mackworth, 2010)

Other concepts include primarily the CWA (Close World Assumption) and the OWA (Open World Assumption); both of these terms are used for knowledge representation (Dutra, 2009), (Matthews, 2006). The CWA is utilized by systems that comprise complete information; these are mostly database applications. For example, we can point to the database of flight companies, which enables us to find a direct flight, such as that from Prague to Vienna. If the database does not include the flight-related data, a clear result will be returned (0 or NULL), and the interpretation is that no direct flight from Prague to Vienna is available.

The OWA is applied if the system contains incomplete information. This concept represents concrete knowledge and indicates how new information can be found. To provide an example, we could refer to patient history systems in the medical field. If a given patient card does not include information about the patient and their allergies, it is not correct to say that the patient does not suffer from allergy now: we cannot establish whether the patient is treated for allergy until additional information to confirm or refute the hypothesis is found.

Generally, we can say that the CWA returns “0” (information missing) and the OWA returns “I do not know”. However, if we investigate the claim in greater detail, we find that it is not entirely true. For example, let us consider the assertion “Ivan is a citizen of the CR”. Another argument (regarded as true) is "a person can be a citizen of one country". If we juxtapose these two situations, everything is in order because Ivan is only a citizen of the CR. If we then have the argument “Ivan is a citizen of the SR”, an error occurs in the CWA system because we mentioned that the person has to be a citizen of only one state. In the given case, the OWA does not generate an error; this system will nevertheless produce the following claim: “If Ivan is a citizen of one state and a citizen of the CR and the SR, then the CR and the SR must be same”. The OWA returns this answer because it uses the UNA. The CWA includes this property but the OWA does not. The system OWA does not use UNA, yet the UNA can be added to the system artificially. The principle is to add disjunction to individual names: in other words, we determine the differences of names.

The main difference between ontologies and databases lies between the OWA and the CWA. From the examples mentioned above, it can be inferred that while ontologies utilize the OWA system of knowledge representation, the CWA is used by databases. A database exploits the UNA for naming entities. Any information missing in a database system has the value of “0”. Any item of information missing in an ontology system is considered unknown.

3 METHOD OF CREATION AND PURPOSE

Another major difference between ontologies and databases is the purpose for which they are created. While ontologies are focused on adding meaning and comprehension, databases concentrate on data storage. In other words, databases are created as effective data warehouses, whereas ontologies are formed for better communication, interoperability, and as the communication bridge between a human and a machine.

Both of these systems use different creation methods. A database system is created from scratch, which means that all tables and their contents are designed in manner indicated here. When we design an ontology system, we attempt to take advantage of existing ontologies or system structuring upon an existing ontology (we thus extend an existing ontology). For example, the SSN ontology is built on the DOLCE upper ontology.

In creating a database system, we apply the normalization of tables; such normalization is used to delete redundant data from the tables and to reduce the complexity. For the given purpose, normal forms are used in the database system; the forms are a set of rules that help to correct the transformation of entities and relationships to the structure of the physical layout of the tables. Today there exist six normal forms, but only the first three are used.

The methodology for the creation of ontology does not include normal forms. A major ontology creation method consists in design patterns. These patterns, however, are not as strict as normal forms: rather than that, they create general rules. In this context, the author recognizes six areas (Ontology patterns 2014):

- Structural pattern
- Syntactic pattern
- Content pattern
- Presentation pattern
- Consideration pattern
- Corresponding pattern

3.1 Ontology creation

Ontology creation is a process comprising several stages. Some of these phases, namely specification, conceptualization, implementation, and maintenance are materialized during the development process; other operations are performed throughout the entire existence of the ontology: data mining, documentation, and evaluation. Some of the main procedural imperatives are described below.

Identify the integration possibility: The framework being applied to build the ontology should allow for some kind of knowledge reuse. In certain cases, integration may involve rebuilding the ontology in a framework different from that where the ontology is available. In some situations, this may be cost-effective, but in others it could be more profitable to

build from scratch a new ontology that perfectly matches the present needs and purposes than to pursue the rebuilding and adaptation of a pre-existent one.

Identify the modules: The modules (building blocks) needed to build a future ontology are identified, that is, the sub-ontologies in which the future ontology should be divided are defined (in integration, the modules are obviously related to ontologies).

Identify the assumptions and ontological commitments: The presented aspects are described in the conceptual model and in the specification requirements document of the future ontology. This is one of the activities where the actual documentation of the ontology can be crucial to facilitate better, faster, and easier reuse. The assumptions and ontological commitments of the building blocks should be compatible with those found for the resulting ontology.

Identify the knowledge to be represented in each module: It is necessary to determine what knowledge should be represented in each building block. At this stage, only an idea is provided of what the modules that will compose the future ontology should be like in order to recognize whether available ontologies are adequate to be reused. A list of essential concepts is identified.

Identify the candidate ontologies: This imperative is subdivided into finding available ontologies and choosing from these the ones that might constitute possible candidates for integration. To choose the candidates, all available ontologies are analyzed according to a series of features. At this stage, only a very general analysis is performed. Some of the features are: strict requirements (the domain, availability, formalism paradigms in which the ontology is available), the main assumptions and ontological commitments, and the main concepts represented. If an ontology does not have adequate values for these properties, it cannot be considered for integration. The properties are used to eliminate ontologies. Other features include desirable requirements or information: where is the ontology available?; at what level is the ontology available?; what kind of documentation is obtainable (such as technical reports or articles)?; where is the documentation accessible? If some of the properties and desirable requirements exhibit appropriate values, then the ontology is a better candidate.

Obtain the candidate ontologies: Getting the desired candidate ontologies includes the processes of their representation and also the acquirement of all available documentation. It is preferable to work with the knowledge level representation of the ontology. However, in most cases, only the implementation level representation is available; we can then use the reengineering procedure or pursue reconstruction via the available documentation.

Study and analyze the candidate ontologies: To fulfill this imperative, we need to perform two activities:

- *Technical evaluation of the candidate ontologies:* It is important to consider certain features, such as: what knowledge is missing; what knowledge should be removed; what knowledge should be reallocated; what knowledge source changes should be performed; what documentation changes should be performed; what definition changes should be made; and what practice changes should be carried out.
- *User assessment of the candidate ontologies:* Special attention has to be paid to the following aspects: the overall structure of the ontology; the distinctions upon which the ontology is built to assess whether they are relevant and required; the relation used to structure the knowledge in the ontology to assess whether it is the desired one; the naming convention rules used to assess whether reuse is simplified and promoted; the quality of the definitions; the quality of the documentation of the ontology; and the knowledge pieces represented.

Choosing the source ontologies: From among the candidate ontologies that passed the strict requirements and those that scored best in the integration-oriented technical evaluation and user assessment, we have to choose the source ontology that best suits our needs and purpose. The best candidate is the one that can be better or more easily adapted to become the baseline ontology. Sometimes more ontologies can be chosen if each one focuses on different elements of the given domain. The choice of source ontologies should be divided into two stages: the first phase, in which one tries to find the candidate ontologies best suited for integration (considering general features, development features, and content features); and the second phase, where it is necessary to tackle the compatibility and completeness of the preliminarily chosen ontologies in relation to the desired resulting ontology.

Apply the integration operations: When the appropriate ontologies to be reused within one particular integration process are found, the knowledge of these ontologies should be integrated. The related integration operations specify how the knowledge from an integrated ontology will be included and combined with the knowledge in the resulting ontology or modified before its inclusion.

Analyze the resulting ontology: After the knowledge integration, one should evaluate and analyze the resulting ontology. Besides exhibiting an adequate design and compliance with the evaluation criteria, the ontology should have an overall uniform level. The resulting ontology should be consistent and coherent all over.

3.2 Database creation

The database creation process comprises several parts, and these parts are used as a tool for better viability and higher performance. Today, although a detailed ontology creation manual exists, we can readily refer to tools that ease the entire procedure. In the case of databases, these instruments are more complex and capable of accelerating the actual creation. The first step is to clarify the type of information that will be stored in the database; to achieve this, we write questions applicable for use by the database. The next step is to verify the answers that will be generated by the database.

When designing a database, it is necessary to rethink the primary and foreign keys. Both these keys simplify the search for data in the database. The primary key is a unique

identifier that has to exhibit the following properties (taken over from *PHP and MySQL*):

- Primary key has to always include some value. (NULL value is not allowed).
- The value has to be the same.
- The value must be unique across the table.

Furthermore, there are properties that the author may not follow. If we, however, comply with these properties, we will improve the database performance and stability.

- The primary key should contain at least a part of the name of the parent table and id expression.
- SQL databases allow the existence of only one primary key in one table. The primary key can be based on multiple columns in the table. However, the combination of these values must be unique.

There are two rules that make it easier to assign a primary key to a particular data column. The first rule says that the column which will be the primary key has to comply with the previous features. The second rule then asserts that if there is no logical primary key, we have to create it ourselves. The second key is a foreign key, and this type is the representative of the primary key of another table. Thus, the primary key in table A is a foreign key in table B; this key is used to interconnect the tables.

Another issue consists in relation, which creates the interconnection of the database tables. The relations can be of three types:

- Relation 1:1 (one to one)
- Relation 1:M (one for many)
- Relation M:N (many to many).

When we have created the database tables, we have to commence the process of normalization. As already mentioned above, normal forms are used to normalize the database. The first normal form (1NF) says that each column in one line must exhibit only one value (it must be atomic). The second normal form (2NF) says that tables correspond to the requirements if they match 1NF. Also, each column in a table whose values are the same in multiple rows is transferred to a separate table. If we intend to normalize according to the third normal form (3NF), we have to normalize the database by 2NF. Database normalization in 3NF requires the following precondition:

- All columns that are not part of the primary key are to be directly dependent on it.

4 WRITING AND EXPRESSION

The difference between a system which includes an ontology and one which comprises a database is in writing the syntax and semantics. A database uses an ER diagram (Entity-relationship model) to describe the syntax; this technique is used for abstract and conceptual data representation. In ontology, however, the syntax is written by logic; the most common is the description logic that corresponds to the OWL DL, a language for creating ontologies.

Thus far we have continuously accentuated the differences between ontologies and databases. However, the two antagonists also share some properties: for example, both of them create the description of a particular section or segment of the real world. A database system includes entities; an entity describes a certain portion of the real world. For example, a relational database comprises entities which are tables in the database. If we refer to a system with an ontology, then an entity is represented by a class. If we proceed somewhat further, we find that the entity contains attributes, and these are the characteristic features of the entity. In comparison with an ontology system, we will see that each class also has its properties; the difference between database attributes and ontology properties consists in dependency. The properties in an ontology are not dependent on a specific class.

Another common element is restriction. Ontologies have axioms that represent the means to describe the equivalence of classes, class disjunction, or the decomposition of classes into subclasses. Databases exhibit restrictions too. For example, a database includes integrity restriction, which guarantees that the stored data are consistent with the defined rules.

5 COMMUNICATION BETWEEN AN ONTOLOGY AND A DATABASE

As already discussed above, there are certain similarities between ontologies and databases. Thanks to these parallel features, we can convert a database to an ontology and vice versa. This type of transfer is possible when the information stored in the ontology corresponds to the data stored in the database. Vysniauskas and Nemuratie describe three possible options to carry the data between these two domains (Vysniauskas and Nemuraite, 1994):

- Using the same conceptual modeling technique to represent the ontology and the database;
- Generating a database schema for the ontology;
- Obtaining a database from the ontology.

5.1 Using the same conceptual modeling technique

This method is not exactly suitable for data conversion; its aim much rather consists in establishing an instrument to simplify the given process. The discussed approach can be described as follows: if the ontology and the database designs apply the same techniques (i.e., conceptual modeling), then the conversion will be easier. Cruz and Brockmans describe a technique that utilizes the UML (Martinez-Cruz et al. 2012), (Brockmans et al. 2006). This simple method is based on using the same UML scheme to design an ontology and a database. However, Brockmans also describes problems which can occur when the method is applied; these problems are mainly related to the use of an object-relational model.

5.2 Obtaining an ontology from a database

This process is currently very popular, and many methods and solutions are available to facilitate data conversion from a relational database**.** Astrova, in the relevant study (Astrova, 2005), named this approach *Ontology Inverse Engineering*.

Generally, these techniques enable access to the data stored in the database.

To define the overall principle of the discussed methods is very simple: it is a process which finds similarities between databases and ontologies. In other words, the given procedure looks for elements which correspond with each other; this operation is essentially expressed in the relationship between classes and tables, attributes and features, and constraints and axioms. These methods are classified as follows:

- Using the database data;
- Using a query analysis;
- Using an analysis of the relational scheme of the database.

This classification shows the type of information known about the database; this information is subsequently used to obtain an adequate ontology.

The following option consists in a reverse procedure, a process aiming to generate the desired database from an adequate ontology; this approach, however, is associated with information loss, which substantially reduces its application possibilities.

6 MOTIVATION FOR USE

Ontologies are needed for the prevention and resolution of communication issues between heterogeneous systems, knowledge sharing, and information fusion; they facilitate information integration and interoperability between heterogeneous knowledge and information sources while maintaining a high level of abstraction.

Typical reasons for the development and use of ontologies (Noy and McGuinness, 2000) are listed in the following summary:

- To share common understanding of the information structure between people or software;
- To enable reuse of the domain knowledge;
- To make the domain assumptions;
- To separate the domain knowledge from the operational knowledge;
- To analyze the domain knowledge.

The motivation for using a database is very simple: people need to store large amounts of data. Here, we are considering amounts of data too large for the internal memory. Moreover, databases provide comfort, security, and facilitate data sharing. The data are thus accessible to a large number of users, and they are stored reliably, with the possibility of reconstruction after an error.

The most typical reasons for the utilization of databases are as follows (Germán, 2004):

- Data dependency;
- Efficient data access;
- Data integrity and security;
- Data administration;
- Concurrent access and crash recovery;
- Reduced application development time.

7 CONCLUSION

A large number of differences and similarities can be traced between ontologies and databases; the aim of this work is therefore to present the most significant examples and, on this basis, improve the understanding of the two concepts.

The first difference to be noted is the actual design. In this respect, a database is created from scratch, while an ontology reuses already existing ontologies. Another point of divergence then lies in the manner of knowledge representation: where an ontology uses the open world assumption, a database relies on the close world assumption.

Table 1. Ontology vs. Database

Item	Ontology	Database
Period of creation	a thousand years ago	a thousand year ago
Knowledge representation	OWA	CWA, UNA
Design methods	using existing ontologies	from scratch
Optimization	ontology patterns	normal forms
Syntax	OWL, RDF languages	entity-relationship model

It is thus evident when we obtain the response to the questions asked. The main differences are summarized in the table 1.

The syntax notion constitutes another area of contrast: ontologies use logic, whereas databases utilize ER-diagrams; then, by further extension, the formal semantic notion is un-important for databases but crucial for ontologies. However, both of these domains also share certain elements. The expressiveness of the domains somewhat resembles the parameters shown in the table below; the rows in the table have the same or a similar meaning.

Table 2. The basic differences

Ontology	Database
classes	tables
properties	attributes
axioms	constrains

But, rather contradictorily to what was indicated above, differences can be found in expressiveness too. A database does not use taxonomy, which nevertheless constitutes a seminal aspect for an ontology.

As already mentioned, the two concepts are substantiated by different reasons for regular usage. Databases are used for clear and safe storing of large amounts of data, while ontologies find application in integrating semantic data or in

communication between heterogeneous systems, and they share the understanding of the structure of information between people or software.

8 ACKNOWLEDGMENT

This work was realized at the CEITEC-Central European Institute of Technology with research infrastructure supported by the project, CZ.1.05/1.1.00/02.0068, financed from the European Regional Development Fund, with the support of the ARTEMIS-JU project "333020 ACCUS-Adaptive Cooperative Control in Urban (sub) Systems"

9 REFERENCES

Astrova, I. 2005. Toward the Semantic Web – An Approach to Reverse Engineering of Relational Databases to Ontologies. *ADBIS research communications*, no. 152.

Brockmans, S., Haase, P., Hitzler, P. & Studer, R.. 2006. A Metamodel and UML Profile for Rule-Extended OWL DL Ontologies. pp. 303. Available from: http://link.springer.com/10.1007/11762256_24

Dutra, L. 2009. Closed-world assumption. Wikipedia: the free encyclopedia [online]. Available from: http://en.wikipedia.org/wiki/Closed-world_assumption [Accessed 2015-01-08]

Germán, D. 2004. Introduction to Database Systems. *TuringMachine* [online]. Available from: http://turingmachine.org/courses/2004/csc370K04/lectures/01_intro_4up.pdf [cit. 2015-01-08]

Martinez-Cruz, C., Blanco, I. & Vila, M.. 2012. Ontologies versus relational databases. *Artificial Intelligence Review*, vol. 38, issue 4, pp. 271-290. Available from: http://link.springer.com/10.1007/s10462-011-9251-9

Matthews, C. 2006. Open-world assumption. *Wikipedia: the free encyclopedia* [online]. Available from: http://en.wikipedia.org/wiki/Open-world_assumption [Accessed 2015-01-08]

Noy, N. & McGuinness, D. 2000. Ontology Development 101: A Guide to Creating Your First Ontology. *Knowledge Systems Laboratory Stanford University* [online]. Available from: http://www.ksl.stanford.edu/people/dlm/papers/ontology-tutorial-noy-mcguinness-abstract.html [Accessed 2015-01-08]

Ontology Design Patterns [online]. 30 May 2014. Available from: http://ontologydesignpatterns.org/wiki/Main_Page [Accessed 2015-01-31]

Poote, D. & Mackworth, A. 2010. Unique Names Assumption. *Artificial inteligence: Foundations of computal agents* [online]. Available from: http://artint.info/html/ArtInt_302.html [Accessed 2015-01-08]

Sir, M., Fiedler, P. & Kaczmarczyk, V.. 2011. Ontology for sensors. *Proceeding CIT'11 Proceedings of the 5th WSEAS international conference on Communications and information technology*, vol. 5, pp. 175-179. Available from: http://dl.acm.org/citation.cfm?id=2028531

Vysniauskas, E. & Nemuraite, L.. 1994. Transforming Ontology Representation from OWL to Relational Database. *Proceedings of the Eleventh International Conference on Machine Learning*, pp. 333-343. Available from: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.109.3036