# Using Ontologies in Database Preservation

Ricardo André Pereira Freitas[1]
José Carlos Ramalho[2]

[1] CLEGI – Lusiada University
Vila Nova de Famalicão – Portugal
[2] Department of Informatics – University of Minho
Braga – Portugal
freitas@fam.ulusiada.pt, jcr@di.uminho.pt

**Abstract.** This paper addresses the problematic Digital Preservation and focuses on the conceptual model within a specific class of digital objects: Relational Databases. Previously, a neutral format was adopted to pursue the goal of platform independence and to achieve a standard format in the digital preservation of relational databases, both data and structure (logical model). Currently, in this project, we intend to address the preservation of relational databases by focusing on the conceptual model of the database, considering the database semantics as an important preservation "property". For the representation of this higher level of abstraction present in databases we use an ontology based approach. At this higher abstraction level exists inherent Knowledge associated to the database semantics that we tentatively represent using "Web Ontology Language" (OWL). We developed a prototype (supported by case study) and define a mapping algorithm for the conversion between the database and OWL. The ontology approach is adopted to formalize the knowledge associated to the conceptual model of the database and also a methodology to create an abstract representation of it.

**Key words:** Digital Preservation, Relational Databases, Ontology, Conceptual Models, Knowledge, XML, Digital Objects

## 1 Introduction

In the current paradigm of information society more than one hundred exabytes of data are used to support information systems worldwide [1]. The evolution of the hardware and software industry causes that progressively more of the intellectual and business information are stored in computer platforms. The main issue lies exactly within these platforms. If in the past there was no need of mediators to understand the analogical artifacts today, in order to understand digital objects, we depend on those mediators (computer platforms).

Our work addresses this issue of Digital Preservation and focuses on a specific class of digital objects: Relational Databases (RDBs). These kinds of archives are important to several organizations (they can justify their activities and characterize the organization itself) and are virtually in the base of all dynamic content in the Web.

### 1.1 Previous Works

In previous work [2] we adopted an approach that combines two strategies and uses a third technique — migration and normalization with refreshment:

- Migration which is carried in order to transform the original database into the new format – Database Markup Language (DBML) [3];
- Normalization reduces the preservation spectrum to only one format;
- Refreshment consists on ensuring that the archive is using media appropriate to the hardware in usage throughout preservation [4].

This previous approach deals with the preservation of the Data and Structure of the database, i.e., the preservation of the database logical model. We developed a prototype that separates the data from its specific database management environment (DBMS). The prototype follows the Open Archival Information System (OAIS) [5] reference model and uses DBML neutral format for the representation of both data and structure (schema) of the database.

### 1.2 Conceptual Preservation

In this paper, we address the preservation of relational databases by focusing on the conceptual model of the database (the information system - IS). For the representation of this higher level of abstraction present in databases we use an ontology based approach. At this level there is an inherent Knowledge associated to the database semantics that we represent using OWL [6].

We developed a prototype (supported by case study) and established an algorithm that enables the mapping process between the database and OWL.

In the following section, we overview the problem of digital preservation, referring to the digital object and preservation strategies. The next section also formulates our hypothesis. In section 3 we overview the relation between ontologies and databases. The prototype and the mapping process from RDBs to OWL is detailed in section 4. At the end we draw some conclusions and specify some of the future work.

## 2 Digital Preservation

A set of processes or activities that take place in order to preserve a certain object (digital) addressing its relevant properties, is one of the several definitions. Digital objects have several associated aspects (characteristics or properties) that we should consider whether or not to preserve. The designate community plays an important role and helps to define

> *"The characteristics of digital objects that must be preserved over time in order to ensure the continued accessibility, usability, and meaning of the objects, and their capacity to be accepted as evidence of what they purport to record"*[7].

### 2.1 The Digital Object

Some distinction can be established between digital objects that already born in a digital context, and those that appear from the process of digitization: analog to digital. In a comprehensive way and encompassing both cases above, we can consider that a digital object is characterized by being represented by a bitstream, i.e., by a sequence of binary digits (zeros and ones) [8].

We can question if the physical structure (original system) of the object is important, and if so, think about possible strategies for preservation at that level (museums of technology). Nevertheless, the next layer — the logical structure or logical object—, which corresponds to the string of binary digits will have different preservation strategies. The bitstream have a certain distribution that will define the format of the object, depending on the software that will interpret it. The interpretation by the software, of the logical object, provides the appearance of the conceptual object, that the human being is able to understand (interpret) and experiment. The strategy of preservation is related to the level of abstraction considered important for the preservation [9]. From a human perspective one can say that what is important to preserve is the conceptual object (the one that the humans are able to interpret). Other strategies defend that what should be preserved is the original bitstream (logical object) or even the original media. Figure 1 shows the relationship between the different levels of abstraction (digital object) and the correspond preservation formats adopted for RDBs in this research.

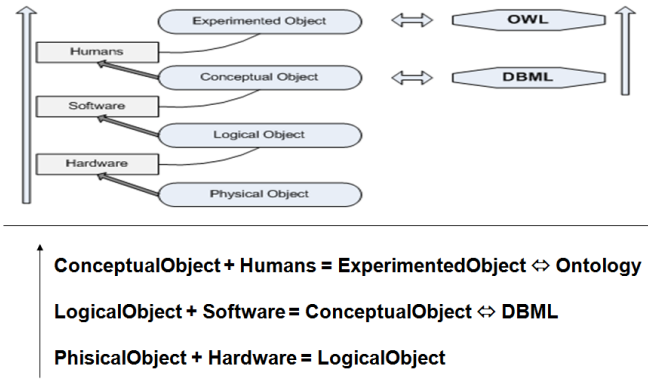


**Fig. 1.** Levels of Abstraction and Preservation Policy

By focusing on a specific class or family of digital objects (relational databases), questions emerge such as: what are the effects of cutting/extracting the object from its original context? Can we do this even when we are referring to objects that are platform (hardware/software) dependent? The interaction between the source of the digital object and the platform results on a conceptual object that

can be different if the platform changes [7]; the output can be different (will the object maintain its original behavior?). The important is the preservation of the essential parts that purport what the object where made for. Either the source or the platform can be altered if what is essential is obtained and also maintaining the meaning of the digital object over long periods of time (long-term scope).

As we mentioned in previous work we address the preservation of the RDBs data and structure by using DBML which ensures that its representation becomes neutral.

### 2.2 Proposed Approach

Our hypothesis concentrates on the potentiality of reaching relevant stages of preservation by using ontologies to preserve of RDBs. This lead us to the preservation of the higher abstraction level present in the digital object, which corresponds to the database conceptual model. At this level there is an inherent **Knowledge** associated to the database semantics (Table. 1). We intend to capture the experimented object (knowledge) through an ontology based approach. The ontology approach is adopted to formalize the knowledge present at the experimented object level and also a methodology to create an abstract representation of it.

**Table 1.** Preservation Policy

| Digital Object | Preservation Levels | Relational Database |
|---|---|---|
| **Experimented Object** | **Ontology** | **Conceptual Model** |
| Conceptual Object | DBML | Logical Model |
| Logical Object | – | Original Bitstream |
| Physical Object | – | Physical Media |

## 3 Ontologies and Relational Databases

There is a direct relation between ontologies and databases: a database has a defined scope and intends to model reality within that domain for computing (even when it is only virtual or on the web); ontology in ancient and philosophical significance means the study of being, of what exists [10].

The (strong) entities present in relational databases have an existence because they were model from the real world: they relate to each other and have associated attributes. In information society and computer science, an ontology establishes concepts, their properties and the relationships among them within a given domain [10].

### 3.1 Ontologies

The study of ontologies in computer science received new impetus due to the growth of the web, their associated semantics and the possibility of extracting knowledge from it. The "Semantic Web" supported by W3C works on establishing a technology to support the *Web of data* [11]. Notice that a tremendous part of the web is based in (relational) databases - specially dynamic information. An ontology can provide readable information to machines [12] at a conceptual level (higher abstraction level). They also enable the integration and interpretability of data/information between applications and platforms.

### 3.2 Database Semantics

A database can be defined as a structured set of information. In computing, a database is supported by a particular program or software, usually called the Database Management System (DBMS), which handles the storage and management of the data. In its essence a database involves the existence of a set of records of data. Normally these records give support to the organization information system; either at an operational (transactions) level or at other levels (decision support - data warehousing systems).

If we intend not only to preserve the data but also the structure of the (organization) information system we should endorse efforts to characterize (read) the database semantics. In other words, we represent the conceptual model of the database using an ontology and intend to preserve that representation.

Ontologies benefit from the fact that they are not platform/system dependent when compared to traditional relational databases.

## 4 From RDBs to OWL

This section presents the work developed to convert databases to ontology, based on a mapping process (mapping algorithm), for preservation. We intend to preserve a snapshot of the database (or a frozen database) by preserving the OWL generated from the database.

We start to briefly refer to some of the related work in this area considering the numerous approaches addressing conversions and mappings between relational databases and ontologies. Then we concentrate our efforts on detailing the mapping process and analyzing the created algorithm. The conducted tests and some of the results are also presented.

### 4.1 Related Work

Several approaches concerning RBDs and ontologies transformations exist and are being addressed continuously. The conversion from databases into an ontology could be characterized as a process in the scope of reverse engineering [13]. While some approaches and works try to establish a mapping language or

a mapping process [14], others use different techniques and strategies for the database translation [15] into an ontology (eg.: OWL).

Considering the Resource Description Framework (RDF) [16] and RDBs, some of the related works, studies and tools are referenced in the W3C Incubator group survey [17]: Virtuoso RDF View; D2RQ; Triplify; R2O; Dartgrid Semantic Web toolkit; RDBToOnto, and others. The extration of ontologies from RDBs are also addressed and referenced in [12].

### 4.2 Mapping Process of RDBs to OWL – Prototype

Our work implements the conversion from RDBs into OWL through an algorithm that performs the mapping process. The developed prototype enables the connection to a DSN (Data Source Name), extracts the data/information needed and gives the initial possibility of selecting the tables of interest (for conversion). It is assumed that the source database is normalized (3NF).

Lets start by enumerating the properties of RDBs that are address and incorporated in the ontology (OWL):

- **Tables** names;
- **Attributes** names and data types;
- **Keys** primary keys, foreign keys (relationships between tables);
- **Tuples** data;

These elements are extracted from the database into multidimensional arrays. Figure 2 shows the arrays structure.

```
tables = Array{ [1] => t1, ... , [n]=> tn }

columns = Array{
                [t1] => Array{
                        [a1] => Array{ [Name] => 'a1_name', [Type] => 'a1_type' },
                        ...
                        [an] => Array{ ... }},
                        ...
                [tn] => Array{...}}

p_keys = Array{
                [t1] => Array{ [a1] => 'pk_t1', ..., [an] => 'pk_t1' },
                ...
                [tn] => Array{...}}

f_key = Array{
                [t1] => Array{
                        [a1] => Array{ [pk_table] => 'tref', [pk_column] => 'tref.aref'},
                        ...
                        [an] => Array{...}},
                ...
                [tn] => Array{...}}

tables_data= Array{
                [t1] => Array{
                        [1] => Array{ [a1]=> 'a1_data', ..., [an] => 'an_data'},
                        ...
                        [m]'=> Array{...}},
                ...
                [tn] => Array{...}}
```

**Fig. 2.** Multidimensional Array Structure

For each `table` on the database we define a **class** on the ontology with the exception of those tables where all attributes constitute a composed primary key (combination of foreign keys). These link tables used in the relational model to

dismount a many-to-many relationship, are not mapped to OWL classes, instead they give origin to **object properties** in the ontology. These object properties have on there domain and range the correspondent classes (database tables) involved in the relationship (Fig. 3).

```
//   Classes (tables) & ObjectProperties (link tables - non_classes)
FOREACH [ table ]
        IF [ ( |columns[table]| = |p_keys[table]| ) AND ( |p_keys[table]| = |f_keys[table])| ) ] THEN
                non_class[] = table
                FOREACH [ columns[table] - 1 ]

                NEW 'objectProperty'
                        Property_Description = 'is_' + f_keys[table][columns[table]][pk_table] + '_of'
                        Domain = f_keys[table][columns[table]][pk_table]
                        Range = f_keys[table][next(columns[table])][pk_table]

                NEW 'objectProperty'
                        Property_Description = 'has_' + f_keys[table][columns[table]][pk_table]
                        Domain = f_keys[table][next(columns[table])][pk_table]
                        Range = f_keys[table][columns[table]][pk_table]

                NEW 'InverseObjectProperties'
                        Property_Description = 'is_' + f_keys[table][columns[table]][pk_table] + '_of'
                        Property_Description = 'has_' + f_keys[table][columns[table]][pk_table]

                END FOR
        ELSE
        class[] = table
        END IF
END FOR
```

**Fig. 3.** Algorithm - Classes and Non Classes

The `foreign keys` of the tables mapped directly to OWL classes also give origin to **object properties** of the correspondent OWL classes (tables). The `attributes` of the several tables are mapped to **data properties** within the analogous OWL classes with the exception of the attributes that are foreign keys (Fig. 4).

The algorithm generates inverse object properties for all relationships among the classes. If the object properties are generated directly from a 1-to-many relationship (which is the last case) it is possible to define one of the object properties as functional (in one direction).

The `tuples` of the different tables are mapped to **individuals** in the ontology and are identified by the associated `primary key` in the database. A tuple in a database table is mapped to an individual of a class (Fig. 5).

The object properties that relates individuals in different classes are only defined in one direction. If in the inverse pair of object properties exists one property that is functional, is that one that it is defined; if not, the generated object property assertion is irrelevant.

In the next table (Fig. 6) we summarize the mapping process. From the conceptual mapping approach and some DBMS heuristics we start to manually convert a relational database (case study database) into OWL using Protégé [18]. The algorithm was then designed based on the defined mapping and from the code analysis (Protégé – OWL/XML format).

### 4.3 Prototype – Tests and Results

The algorithm was then tested with the case study database. Figure 7 shows the database logical model and the ontology conceptual approach. It was nec-

```
//   Sub Classes of Thing & Disjoint all & Object and Data Properties
class_disjoint[] = class
FOREACH [ class ]
        NEW class 'SubClassOf' owl:Thing
        FOREACH [ class_disjoint ]
                IF  [ class IN class_disjoint ] THEN
                        NEW 'DisjointClasses'
                                Class_Description = class
                                Class_Description = class_disjoint
                END IF
        END FOR
        pop(class_disjoint)

        FOREACH [ f_keys[table] as fk ]
                NEW 'ObjectProperty'
                        Property_Description = 'is_' + fk['pk_table'] + '_Of'
                        Domain = fk['pk_table']
                        Range = class
                NEW 'ObjectProperty'
                        Property_Description = 'has_' + fk['pk_table']
                        Domain = class
                        Range = fk['pk_table']
                NEW 'InverseObjectProperties'
                        Property_Description = 'is_' + fk['pk_table'] + '_Of'
                        Property_Description = 'has_' + fk['pk_table']
                NEW 'FunctionalObjectProperty'
                        Property_Description = 'is_' + fk['pk_table'] + '_Of'
        END FOR

        FOREACH [ columns[table] as table_data ]
                IF [ f_keys[$table][table_data['Name']]['pk_column'] != table_data['Name'] ] THEN
                        NEW 'DataProperty'
                                Property_Description = 'has_' + table_data['Name']
                                Domain = class
                                Range = data_type
                END if
        END FOR
END FOR
```

**Fig. 4.** Algorithm - Structure Genaration

```
//   tuples -> Individuals   //
FOREACH [ class ]
        FOREACH [ tables_data[table] as tuple ]
                primary_key = class
                FOREACH [ p_keys[table] as pk)
                        primary_key = primary_key + pk
                END FOR

                NEW 'ClassAssertion'
                        Class_Description = class
                        NamedIndividual = primary_key
                FOREACH [ tuple as kt=>t ]
                        IF [ NOT [ kt IN array_keys(f_keys[table]) ] ]
                                NEW 'DataPropertyAssertion'
                                        DataProperty = class + '_has_' + kt
                                        NamedIndividual = primary_key
                                        Literal = t
                        ELSE
                                NEW 'ObjectPropertyAssertion'
                                        ObjectProperty = f_keys[table][kt]['pk_table']
                                        NamedIndividual = primary_key
                                        NamedIndividual = f_keys[table][kt]['pk_table'] + '_' + t
                        END IF
                END FOR
        END FOR
END FOR
//   tuples -> ObjectProperties (link tables)   //
FOREACH [ non_class ]
        FOREACH [ columns[table] - 1 ]
                FOREACH [ tables_data[table] as tuple ]
                NEW 'ObjectPropertyAssertion'
                                ObjectProperty = f_keys[table][columns[table]]['pk_table']
                                NamedIndividual = f_keys[table][next(columns[table])]['pk_table'] +
                                                '_' + tuple[f_keys[table][next(columns[table])]['pk_column']]
                                NamedIndividual = f_keys[table][columns[table]]['pk_table'] +
                                                '_' + tuple[f_keys[table][columns[table]]['pk_column']]
                END FOR
        END FOR
END FOR
```

**Fig. 5.** Algorithm - Individuals

| RDB | OWL |
|---|---|
| Tables | Classes |
| *If (#attributes = #primary keys = #foreign keys) → Object Property* | |
| Foreign Keys | Object Properties |
| Primary Keys | Individuals Identification |
| Other Attributes | Data Properties |
| Tuples | Individuals |
| | • Inverse Object Properties Generation<br>• Functional Object Properties Definition<br>• Disjoin All Classes |

**Fig. 6.** Mapping Process Sumarized

essary to do some adjustments in order to achieve a consistent ontology. Then we successfully use the HermiT 1.3.3 reasoner [19] to classify the ontology. The inverse "object properties assertions" that the algorithm do not generates for the individuals were inferred. Some equivalent (and inverse functionality) object properties were also inferred.

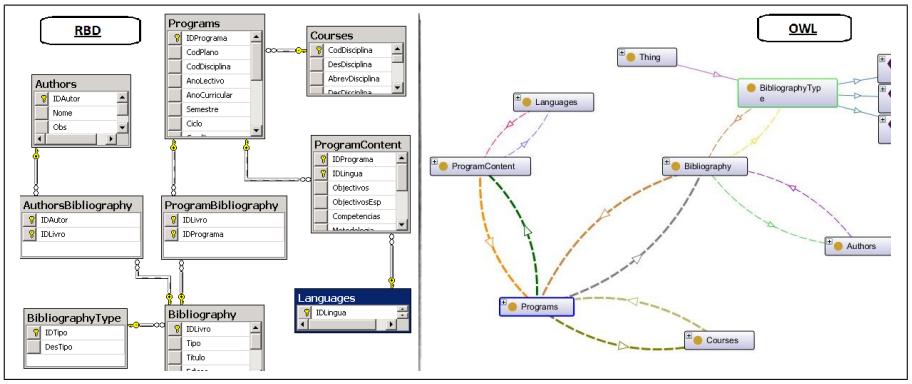


**Fig. 7.** RDB Logical Model *vs* Ontology Overview

The next step consisted on testing the algorithm with other databases. We use one MySQL database and two MSSQL Server databases (the maximum tables size were about tens of thousands records). All databases used in this research are from the University Lusiada information system.

The results were very satisfactory because the algorithm achieve similar results of the ones obtained with the case study database only with minor inconsistencies. The processing time is an issue directly related to the dimension of the database (it is necessary to test the algorithm with huge databases [millions of records] in machines with powerful processing capability).

## 5 Conclusion and Future Work

Ontologies and databases are related to each other because of their characteristics. Using ontologies in database preservation is an approach to capture the "knowlodge" associated to the conceptual model of the database.

In previous work we preserve the database data and structure (logical model) by ingesting the database in a XML based format into an OAIS based archive.

Here, we present the work developed in order to convert databases to ontology, based on a mapping process (mapping algorithm), for preservation. In order to preserve a snapshot of the database (or a frozen database) we preserve the ontology (OWL, also a XML based format) obtained from the application of developed algorithm to the source database. We tested the algorithm with few

databases and the results were acceptable in terms of consistency of generated ontology (and comparing to the results obtained with the case study database).

This generated ontologies will induce the development of a new database browser/navigation tool.

Ontologies also have other potentialities such as the asset of providing answers to questions that other standards are limited. For example, in terms of metadata, one issue that we intend to also address in future work.

We also anticipate the possibility of integration between OWL Web Ontology Language [6] and Semantic Web Rule Language (SWRL) [20] to consolidate the asserted and inferred knowledge about the database and its information system.

## References


1. Pat Manson, "Digital Preservation Research: An Evolving Landscape," European Research Consortium for Informatics and Mathematics - NEWS, 2010.
2. R. Freitas, J. Ramalho, "Relational Databases Digital Preservation," Inforum: Simpósio de Informática, Lisboa, Portugal, 2009, ISBN: 978-972-9348-18-1; [Online]. Available: http://repositorium.sdum.uminho.pt/handle/1822/9740
3. M. Jacinto, G. Librelotto, J. Ramalho, P. Henriques, "Bidirectional Conversion between Documents and Relational Data Bases," 7th International Conference on CSCW in Design, Rio de Janeiro, Brasil, 2002.
4. Ricardo Freitas, "Preservação Digital de Bases de Dados Relacionais," Escola de Engenharia, Universidade do Minho, Portugal, 2008
5. Consultative Committee for Space Data Systems. "Reference Model for an Open Archival Information System (OAIS) - Blue Book," National Aeronautics and Space Administration, Washington, 2002.
6. "OWL - Web Ontology Language" [Online]. Available: http://www.w3.org/TR/owl-features/
7. A. Wilson, "Significant Properties Report," InSPECT Work Package 2.2, Draft/Version 2 (2007), [Online]. Available: http://www.significantproperties.org.uk/documents/wp22_significant_properties.pdf.
8. Miguel Ferreira, "Introdução à preservacao digital - Conceitos, estratégias e actuais consensos," Escola de Engenharia da Universidade do Minho, Guimarães, Portugal, 2006.
9. K. Thibodeau, "Overview of Technological Approaches to Digital Preservation and Challenges in Coming Years," presented at The State of Digital Preservation: An International Perspective, Washington D.C., 2002.
10. Tom Gruber, "Ontology," Entry in the Encyclopedia of Database Systems, Ling Liu and M. Tamer Ozsu (Eds.), Springer-Verlag, 2008.
11. http://www.w3.org/standards/semanticweb/
12. H. Santoso, S. Hawa and Z. Abdul-Mehdia, "Ontology extraction from relational database: Concept hierarchy as background knowledge," Knowledge-Based Systems, Elsevier, 2010
13. C. He-ping, H. Lu, C. Bin, "Research and Implementation of ontology automatic construction based on relational database," International Conference on Computer Science and Software Engineering. IEEE Computer Society, 2008.
14. I. Myroshnichenko , M. C. Murphy, "Mapping ER Schemas to OWL Ontologies," Proceedings of the 2009 IEEE International Conference on Semantic Computing, p.324-329, September 14-16, 2009

15. K. M. Albarrak , E. H. Sibley, "Translating relational & object-relational database models into OWL models," Proceedings of the 10th IEEE international conference on Information Reuse & Integration, Las Vegas, Nevada, USA, 2009
16. http://www.w3.org/RDF/
17. "A Survey of Current Approaches for Mapping of Relational Databases to RDF," W3C Incubator Group, 2009
18. http://protege.stanford.edu
19. http://hermit-reasoner.com/
20. "SWRL: A Semantic Web Rule Language Combining OWL and RuleML" [Online]. Available: http://www.w3.org/Submission/SWRL/
21. XML, "Extensible Markup Language", in W3C - The World Wide Web Consortium [Online]. Available: http://www.w3.org/XML/