Quality of Life through Quality of Information
J. Mantas et al. (Eds.)
IOS Press, 2012

doi:10.3233/978-1-61499-101-4-38

Data Definition Ontology for clinical data integration and querying

Ariane ASSELE KAMA[a,1], Audi PRIMADHANTY[a], Rémy CHOQUET[a], Douglas TEODORO[b], Frank ENDERS[c], Catherine DUCLOS[d] and Marie-Christine JAULENT[a]
[a]*INSERM, UMR_S 872, Eq. 20, Université Pierre et Marie Curie, Paris, France;*
[b]*SIMED, University of Geneva and University Hospitals of Geneva, Switzerland;*
[c]*Averbis, Freiburg, Germany;*
[d]*LIM&BIO, Université Paris XIII, France;*

Abstract. This paper describes an approach to build a Data Definition Ontology (DDO) in the context of full domain ontology integration with datasets in order to share and query clinical heterogeneous data repositories. We have adapted an existing semantic web tool (D2RQ) to implement a process that automatically generates the DDO from a database information model, thanks to reverse engineering and schema mapping approaches. This study has been performed in the context of the DebugIT European project (Detecting and Eliminating Bacteria UsinG Information Technology) that aims to control and monitor the bacterial growth via a semantic interoperability platform (IP). The evaluation of the process is based, first, on the accuracy of the produced DDO for different samples of database storage and second, by checking the congruency between the DDO and the D2RQ database mapping file.

Keywords. D2RQ, Ontology, Database information model, Database integration and mapping

Introduction

Nowadays, semantic interoperability is a broadly used paradigm to approach the problem of sharing data from heterogeneous data sources. Many semantic interoperability platforms have been developed in various projects to apply this paradigm for data analysis by querying heterogeneous and distributed data sources (*e.g.*DebugIT [1], EHR4CR [2]). In such systems, the semantics of data have to be defined in order to achieve their integration and querying,using for instance ontologies. In this case, a semantic correspondence is explicitly described between one or more data sources and the shared domain ontologies available in the platform [3, 4]. The integration of domain ontology with datasets is an uneasy task due to the "semantic gap" existing between the data and the domain ontology [5]. Our proposition is to formalize the knowledge about the data in an ontological formalism in order to simplify the mapping between data and domain ontology (knowledge about the domain). In this context, we introduce the notion of data definition ontology (DDO) to formalize conceptually a data source. The key aspect of a DDO is to express the structure as well

[1] Corresponding Author. Ariane Assélé Kama, INSERM UMRS_872 eq 20, Centre de recherche des cordeliers, 15 Rue de l'école de médecine, 75006 Paris, France ; Email: ariane.asselekama@gmail.com.

as the vocabulary of the underlying database schema in RDF [6]. The purpose of this study is to automate the DDO generation using reverse engineering and schema mapping approaches [7-9]. The present work has been performed in the context of the EU FP7 DebugIT project, which aims to monitor bacterial resistance evolution in a European-wide context. The DebugIT framework aggregates clinical data stored in different hospitals through a semantic interoperability platform dedicated to query distributed semantic endpoints. Thereby, the data definition ontology has been used to achieve full domain ontology integration with datasets and helps the data queries to be setup [10].

1. The data integration process within the DebugIT Project

In order to integrate properly the clinical databases with other European peers, some steps have to be taken to normalize the data and ensure their quality. These clinical databases are published on the Web as SPARQL endpoints [11] following these steps:

- **Getting data**. Even though large amounts of data that can be shared might exist, they are not always readily available to be used in the project.
- **Normalizing data**. The data should be normalized against commonly used medical terminologies (*e.g.* ATC[12] for drugs, NEWT[13] for bacteria).
- **Generating the database mapping file**. The database content is mapped to RDF by a declarative mapping, which specifies how resources are identified and how property values are generated from database content.Each database component is identified by an URI (Uniform Resource Identifier). These connections are defined in a "mapping file" (D2RQ mapping file).**Setting up a SPARQL endpoint**. SPARQL endpointshave been built in the DebugIT project for each clinical database, using the semantic web tool D2R Server and the D2RQ mapping file. This tool is usedfor publishing relational databases on the Web as RDF triples, enabling full SPARQL query capabilities [14].
- **Generating the Data Definition Ontology (DDO).**The last step of the data integration process corresponds to the workdescribed in this paper. We define a process to automatically extractthe corresponding conceptual view (the DDO) from a relational database schema. In the project the DebugIT Core Ontology (DCO) is used throughout an interoperability platform to translate the datasets into common and shared knowledge. To be able to achieve full domain ontology integration (DCO) with datasets, the DDO isdesigned to bridge the semantic gap between the "real world data" and formal ontologies, and to help the data queries to be setup (manually or automatically). As a result, some rules have been defined in the project to align the DDO's to the DCO. These rules consist in aligning fields from databases to domain concepts (concept matching rules) and mapping the vocabulary of the database (instances) to domain concepts (vocabulary matching rules).

2. The Data Definition Ontology (DDO) implementation process

The overarching goal of the study is to define the rules that will be used for mapping the database schema to the ontology. In the early stages of the DebugIT project, the DDO wasbuilt manually. It was definedby the following characteristics:

- The DDO describes the elements of a database schema using the RDF/S – OWL syntax and respecting the N3 language. TheSPARQL query language is used to query the database using the DDO concepts.
- A table within the data model is represented as a concept that is identified with a distinct URI.The same is done for each column within the database model, which is not a structural element (*e.g.* a foreign key).
- Relationships between the tables within the data model are getting represented as object properties.

As specified in the ISO/IEC 11179 Metadata Registry standard [15], each DDO data element is *uniquely identified* (*e.g.*by a specific URI), *named* (*e.g.* using the Upper camel case) and *defined* (*e.g.* using the SKOS notation), and *classified* in a classification scheme (*e.g.*ontology). Predefined mapping rules are available to build an ontology from a relational database schema, as described in some existing approaches like reverse engineering, schema mapping and data mining [8,9,16]. Such rules have been reused and adapted to define new ones and propose a generic method to automatically create a DDO from a database schema [17]. We have implemented these rules within the D2RQ semantic web tool, which uses the D2RQ mapping language for mapping databases schemas to RDF vocabularies and OWL ontologies [18]. The mapping process starts by detecting particular cases for tables in the database schema to map each database component (*e.g.* table, column, constraint) to the corresponding ontology component (*e.g.* class, property, relation).During the DDO generation process, we are able to assess the completeness of the database, calculating the fill rate of each database component. This is an important information that can be later used to identify important properties or in a quality process. In the project, the process was applied for MySQL and Oracle databases, but can also be used with other databases as long as they have been previously supported by the D2RQ platform (*e.g.* PostgreSQL, SQL, Interbase etc.)

In MySQL database case, we have defined two rules to identify foreign keys when they are not clearly specified.First, we check if the index column name is similar to a table name. Second, we consider that a *primary key used in another table is a foreign key*. However, in cases where an important difference exists between a column name and the potential table name in which this column is considered as primary key, the "string" comparison cannot work. For instance, table*"eoc"* with columns *"id, date, patient"* and table *"exam"* with columns *"id, date, service, episode_of_care_id"*, applying the previous rules could not identify *"episode_of_care_id"* as foreign key in *"exam"* table. Another way to detect a foreign key is to see whether there is a table name that is a subset of the index column name (*e.g.* index column "fk_bacteria" will be detected as a foreign key to table "bacteria"). Again, in cases where the column's name does not reflect the table name, this foreign key will not be detected.In ORACLE database case, we faced some issues due to the generation by ORACLE of a lot of unused tables. To overcome this issue, we have defined a variable allowing the user to exclude or include tables.

3. Results

Initially in the DebugIT project, the DDO was manually built. In the present study, the DDO automatic generation process was applied for two types of database storage corresponding to clinical databases from the project partners. For each database schema, the resulting ontology was then getting compared to the previous one, which was created manually.As an example, the table *Culture_Results(id, culture_id, Identified_bacteria, Antibiotic_Tested, Result, CMI)*is transformed in the following ontological corresponding component:

```
ddo:CultureResults a rdfs:Class;
    rdfs:isDefinedBy <http://ddoURI>;
    rdfs:label “””table_label_EN”””@en;
    skos:definition “””concept definition English”””@en;
    rdfs:subClassOf
        [ a owl:Restriction; owl:onProperty ddo:hasId; owl:someValuesFrom xsd:int; ddo:completeness
        [quex:percentage 100.0]],
        [ a owl:Restriction; owl:onProperty ddo:CultureID; owl:someValuesFrom ddo:Culture],
        [ a owl:Restriction; owl:onProperty ddo:hasIdentifiedBacteria; owl:someValuesFrom ddo:Bacteria],
        [ a owl:Restriction; owl:onProperty ddo:hasAntibioticTested; owl:someValuesFrom ddo:Drug],
        [ a owl:Restriction; owl:onProperty ddo:hasResult; owl:someValuesFrom ddo:Sensitive],
        [ a owl:Restriction; owl:onProperty ddo:CMI; owl:someValuesFrom xsd:float].
```

After building the DDO, each database table/column will be automatically mapped to their corresponding DDO concepts in the D2RQ mapping file. In the DebugIT project, SPARQLendpoints were being set up to make the data accessible remotely and seamlessly via the web. In order to correctly use the DDO and the database mapping file (D2RQ mapping file) in the querying process, the DebugIT team has developed a set of “validation rules”. Practically, it enables to check the congruency between the DDO and the D2RQ mapping file. The set of validation rules aims to identify errors between the two files in these specific cases:

- The D2R mapping file maps “data” to DDO concepts (classes or properties) that do not exist in the DDO. When the user queries the database, using DDO concepts, these data could never been retrieved.
- The DDO describes concepts that are not mapped to data in the D2R mapping file. The queries based on these concepts will never yield results.

4. Discussion

We are aware that the DDO building processcannot be entirely automated. Some types of information cannot be foreseen before the DDO generation. For example, some data in the database could refer to external terminologies or standard, which are not specified as column “type”. Practically, the user will have to “manually” specify this information. The initial manual creation of the DDO defines all columns that are not structural elements (*e.g.* int, float, datetime, etc.) as "concept". Additionally, thenon-structural columns in the automation process are not systematically defined as "concepts", but rather as“property”which value is a "concept". In the current version of the process, all the tables / columns are automatically generated as concepts or properties in the context of MySQL. In the case of ORACLE, we propose a first approach allowing the user to select tables according to his needs. This approach is actually an interesting perspective for the application to “semi-automate” the DDO

generation. Consequently, the user is able to (1) select tables/columns, (2) specify external terminologies or standards (3) specify foreign keys and (4) choose or modify concepts and properties names. These perspectives could be implemented through a graphical interface.

Acknowledgement. DebugITis funded by the EU Seventh FP (ICT-2007.5.2-217139) which is gratefully acknowledged. We also thank the input of our project partners.

References

[1] http://www.debugit.eu/ (last access May 2012)
[2] http://www.ehr4cr.eu/ (last access May 2012)
[3] Wache H, Vogele T, Visser U, Stuckenschmidt H, Schuster G, Neumann H, Hubner S. Ontology-based integration of information - a survey of existing approaches. In Stuckenschmidt H. ed. IJCAI-01 Workshop: Ontologies and Information Sharing. 2001; 108-117
[4] Ghawi R, Cullot N. Database-to-Ontology Mapping Generation for Semantic Interoperability. Third International Workshop on Database Interoperability (InterDB 2007), held in conjunction with VLDB 2007. Vienna, Austria: 2007.
[5] Schulz S, Schober D, Daniel C, Jaulent MC. Bridging the semantics gap between terminologies, ontologies, and information models. Stud Health Technol Inform. 2010; 160(Pt 2):1000-4
[6] http://www.w3.org/RDF/ (last access May 2012)
[7] Gomez-Pérez A, Fernandez-Lopez M, Corcho O. Ontology Development Methods and Methodologies. Ontological Engineering, Springer Verlag, 113-153. Madrid, Espagne. 2004.
[8] Ding L, Kolari P, Ding Z, Avancha S, Finin T, Joshi A. Using Ontologies in the Semantic Web: A Survey. In TR-CS-05-07 UMBC. 2005.
[9] Sahoo SS, Halb W, Hellmann S, Idehen K, Thibodeau T, Auer S, Sequeda J, Ezzat A. A Survey of Current Approaches for Mapping of Relational Databases to RDF. 2009.
[10] Lovis C, Colaert D, Stroetmann VN. DebugIT for patient safety - improving the treatment with antibiotics through multimedia data mining of heterogeneous clinical data. Stud Health Technol Inform. 2008 ; 136: 641-6
[11] http://www.w3.org/TR/rdf-sparql-query/ (last access May 2012)
[12] http://www.whocc.no/atcddd/ (last access May 2012)
[13] http://www.uniprot.org/taxonomy/ (last access May 2012)
[14] Bizer C, Cyganiak R. D2R Server - Publishing Relational Datbases on the Semantic Web. Poster at the 5th International Semantic Web Conference (ISWC2006). 2006.
[15] http://metadata-stds.org/11179/ (last access May 2012)
[16] Cullot N, Ghawi R, Ytongnon K. DB2OWL: A Tool for Automatic Database-to-Ontology Mapping. Université de Bourgogne. 2007.
[17] http://debugit3.spim.jussieu.fr/rules_ddo.html (last access May 2012)
[18] http://d2rq.org/ (last access May 2012)