Upgrading Relational Legacy Data to the Semantic Web

Jesús Barrasa Rodríguez
Ontology Engineering Group
Dept. Inteligencia Artificial. Fac. Informática
UPM Spain
+34 91 3367467
jbarrasa@eui.upm.es

Asunción Gómez-Pérez
Ontology Engineering Group
Dept. Inteligencia Artificial. Fac. Informática
UPM Spain
+34 91 3367467
asun@fi.upm.es

ABSTRACT
In this poster, we describe a framework composed of the R_2O mapping language and the ODEMapster processor to upgrade relational legacy data to the Semantic Web. The framework is based on the declarative description of mappings between relational and ontology elements and the exploitation of such mapping descriptions by a generic processor capable of performing both massive and query driven data upgrade.

Categories and Subject Descriptors
H.2.8 [Database applications]

General Terms: Languages, Management.

Keywords: Semantic Web, Relational Databases, Upgrade, Database-to-Ontology mappings.

1. INTRODUCTION
The problem tackled in our proposal is **the association of explicit semantics -based on existing ontologies- with the content of legacy databases** (conforming the "Deep Web"[2]) to facilitate the interchange, combination, integration among systems & processes or automatic reasoning on their content. In other words, its **upgrade to the Semantic Web**.

Table 1. Upgrade approaches classification criteria

	Approach	Pros / Cons
Ontology	Created ad-hoc	☺Simpler mapping situations. ☺Less expressivity required. ☹Less flexible. ☹Only intended use.
	Reuse existing one	☺Promotes reuse. ☹More expressivity required to overcome differences ☺Different semantic views for the same database.
Architect.	Wrapper	☹Non reusable ☹Complex evolution and maintenance ☺Better performance
	Generic engine + declarative definition	☺Simple evolution and maintenance ☺Explicit mappings ☺Allows composition and verifications ☹Lower performance
Exploitation	Massive upgrade (batch)	☺Creates independent repository, no interference with local processing at source ☺High query performance ☹Inadequate if rapidly changing data source.
	Query driven (on demand)	☺Fresh information. Adequate for changing data sources. ☹Delay in query processing ☹Competes with local processing at source

This approach finds its inspiration in Tim Berners Lee's claim back in September 1998: *"one of the main driving forces fo the Semantic Web will be the expression on the Web, of the vast amount of relational database information in a way that can be processed by machines"*[1].

Table 1 describes the pros and cons of the main approaches to database data upgrade.

2. WORK OBJECTIVES
The framework we present in this poster is intended to:

- Be capable of mapping independently conceived, developed and maintained ontologies and databases.
- Implement the more flexible architecture: a generic engine and exploiting a declarative definition of correspondences.
- Allow massive (batch) and query driven upgrade execution.
- Provide an extendable set of primitives, not limited by DBMS expressivity.

The following diagram describes our proposal schematically.

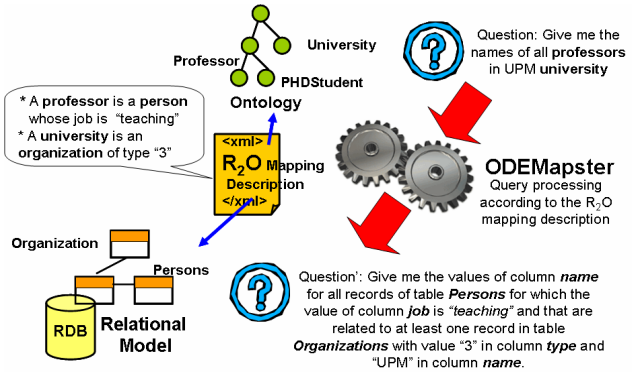

Figure 1. Schematic description of the proposal

3. MAIN FEATURES OF OUR APPROACH.

3.1 The R_2O language
R_2O is a declarative, XML-based language that allows the description of arbitrarily complex mapping expressions between ontology elements (concepts, attributes and relations) and relational elements (relations and attributes). The strength of the R_2O language lies in its expressivity and in its DBMS independence. The elements of the language providing such qualities are **conditions** & **operations** and the **rule-style mapping definition for attributes**.[1]

3.1.1 Conditions and Operations
Conditions and operations allow the description of *"under which circumnstances a database individual (a relational tuple, a database record) can be upgraded to a Semantic Web individual (an instance of the target ontology)"* and *"what kind of*

[1] A complete description of the R_2O Language is available in [3]

transformations are needed to create a Semantic Web individual from a database individual" respectively. Both are defined in terms of an extendable set of primitives and are identified by their names and the set of named parameters they accept. The values of such parameters can be constant values (**has-value**), variables referring record fields from the database (**has-column**), or the result of the execution of other operations (**has-transform**).

The first R_2O excerpts describe a condition based on the *"match-regexp"* primitive. The condition is verified if the content of column *salaryRange* of table *jobs* matches the regular expression.

```
condition "match-regexp"
    arg-restriction
      on-param "string"
      has-column jobs.salaryRange
    arg-restriction
      on-param "regexp"
      has-value ([:digit:]*)-([:digit:]*)
```

The second fragment describes an operation based on the *"concat"* primitive. The operation concatenates two constant strings with the content of column *id* of table *jobs*.

```
operation "concat"
    arg-restriction
      on-param "string1"
      has-value "http://net.testing.r2o/job-"
    arg-restriction
      on-param "string2"
      has-transform
        operation "concat"
          arg-restriction
            on-param "string1"
            has-column jobs.id
          arg-restriction
            on-param "string2"
            has-column jobtypes.code
```

Other primitives defined in the first version of the language are: *plus, minus, multiply, divide, apply-regexp, in-keyword, hi-tan, lo-than, equals, hieq-than, loeq-than, etc.*

3.1.2 Attribute mapping definitions

Mapping definitions for attributes are defined as sets of *if-then* rules that allow the conditional generation of attribute values as well as multivaluation. The structure of an attribute mapping definition is described by the following example. The value of the ontology attribute *type* is calculated based on the application of the set of rules (**selector**): If the condition part (**applies-if**) is verified, then the action part (**aftertransform**) is executed to generate a value.

```
attributemap-def"http://net.testing.r2o/jobs#type"
    selector
      applies-if
        condition […condition desc 1…]
      aftertransform
        operation […transformation desc 1…]
    selector
      applies-if
      aftertransform …
```

3.2 The ODEMapster processor

The ODEMapster processor generates Semantic Web instances from relational instances based on the mapping description expressed in an R_2O document. ODEMapster offers two modes of execution: **Query driven upgrade** (on-the-fly query translation) and **massive upgrade batch process** that generates all possible Semantic Web individuals from the data repository.

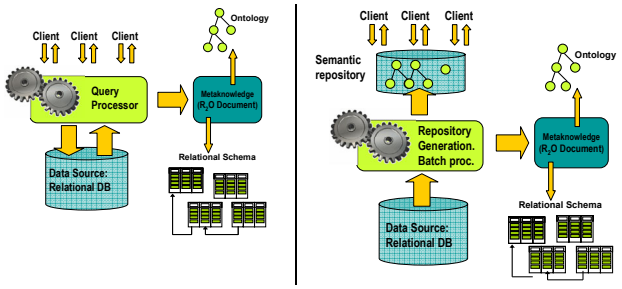

Figure 2. ODEMapster execution modes

The operations of ODEMapster are not limited by the expressivity of the DBMS. The set of primitives can be extended with delegable or non delegable primitive conditions and operations. The processor will delegate the execution of certain actions to the DBMS and execute the rest itself (post processing). The main steps of its executions are: Query & R_2O parsing, SQL generation, SGBD execution result grouping and finally post-processing.

4. RELATED WORK

The definition and exploitation of mappings between relational databases and ontologies have been dealt with in the area of information integration with approaches like OBSERVER[8], PICSEL[4], MOMIS[5], all of them wrapper dependent and following a mediator approach (no complex mapping situations). Other upgrade approaches are D2R[6], KAON-Reverse[7], which are less expressive and only allow massive batch upgrade.

5. CONCLUSSIONS AND FUTURE WORK

Achievements: R_2O definition and implementation of the first version of the ODEMapster processor. Tested in the generation of the semantic portal www.esperonto.net/fundfinder in the context of the ESPERONTO project. Currently being tested and soon available for download at www.oeg-upm.net.

Future trends: Validation of ontology axioms. Ontology learning based on database instance data. Mappings evolution, change propagation. Evaluation and enhancements on performance.

6. REFERENCES

[1] Tim Berners Lee. *Relational databases on the semantic web*. Design Issues (published on the Web), September 1998. http://www.w3.org/DesignIssues/RDB-RDF.html.

[2] Michael K. Bergman. *The deep web: Surfacing hidden value*. The Journal of Electronic Publishing, 7(1), August 2001.

[3] Jesús Barrasa et al. *R_2O, an extensible and semantically based database-to-ontology mapping language*. In Proceedings of the 2nd Workshop on Semantic Web and Databases, Toronto, (Can) Aug.04.

[4] Francois Goasdoue et al. *The use of CARIN language and algorithms for information integration: The PICSEL system.* International Journal of Cooperative Information Systems, 2000.

[5] S. Bergamaschi et al. *Semantic integration of semistructured and structured data sources*. SIGMOD Rec., 28(1):54–59, 1999.

[6] Christian Bizer. *D2RMap - a database to rdf mapping language*. In 12th Intl World Wide Web Conference Budapest May 2003.

[7] L. Stojanovic et al. *A reverse engineering approach for migrating data-intensive web sites to the semantic web*. In Proceedings of the IIP-2002 (IFIP WCC2002), Montreal, Canada, 2002.

[8] E. Mena et al.*OBSERVER: An Approach for Query Processing in Global Information Systems based on Interoperation across Pre-existing Ontologies.* International Journal on Distributed and Parallel Databases 8(2):223-271. April 2000