Rewritting Data Cleaning Operations Defined at a Conceptual Level

RICARDO ALMEIDA, PAULO MAIO, PAULO OLIVEIRA, ISEP-IPP –
SCHOOL OF ENGINEERING – POLYTHECNIC OF PORTO
JOÃO BARROSO, INESC TEC AND UNIVERSITY OF TRÁS-OS-MONTES E ALTO DOURO

The handling of increasing amounts of data creates the need to deal with redundant and/or complementary repositories which are disparate in their data models and/or their data structures. Current data cleaning techniques developed to tackle data quality problems are just suitable for scenarios were all repositories share the same model and structure. Recently, a novel methodology was proposed to overcome this limitation. In the context of this methodology, it is outlined a generic process aimed to rewrite Data Cleaning Operation (DCO) specified at a conceptual level for a given domain ontology, and whose structure and semantics are described by the E-DQM ontology, according to another data structure of a target repository and another vocabulary required by an existing Data Cleaning Tool. The proposed algorithm includes validation of the rewritten DCOs and its serialization to the required output format.

• **Information Quality→ Data Cleaning**

Additional Key Words and Phrases: Data Cleaning, Rewriting Process, Ontology, Vocabulary, Schema.

1. INTRODUCTION

As a result of several advents, such as the Globalization [Redman 2001], the Internet of Things (IoT) [Atzori et al. 2010] and the Big Data [Singh and Singh 2012] and [Snijders et al. 2012], and their consequent technological advances, organizations are being led to generate and to collect an ever increasing volume of data. Thus, organization information systems are being constantly challenged to adapt and to integrate several and disparate data sources (or repositories), which (may) vary in two dimensions: (i) the subjacent data model (e.g. relational data model [Codd 1970], object-oriented [Booch 1993], document-oriented [Han et al. 2011], triples/graphs [Han et al. 2011]); and (ii) the data structure (or schema). The latter is seen as a conceptualization of a given domain/application (e.g. health, business transactions, sports) and, therefore, may differ on terminology, semantics and granularity.

In this context of change, for the same domain (or application), organizations usually end up with more than one repository with redundant and/or complementary information. Moreover, data of a single repository is often (i) collected under different circumstances (e.g. application/device used to collect the data and, therefore, the applied business rules); and/or (ii) as result of one (or more) data integration processes; both leading to Data Quality Problems (DQPs). Within this work, DQPs are problems that exist at data level, such as: missing values in mandatory attributes; domain violations; uniqueness violations; business rules violations; existence of duplicates, which can be equal, almost equal, or even inconsistent [Oliveira et al. 2005]. These problems are usually addressed through a data cleaning technique (or process) that requires a domain expert to specify Data Cleaning Operations (DCOs) targeted to first detect DQPs and further to correct them [Milano et al. 2005] and [Dasu et al. 2003].

Through an extensive analysis of the literature on data cleaning techniques [Fürber and Hepp 2011] and [Oliveira et al. 2009] and [Weis and Manolescu 2007] and [Knuth and Sack 2014], it is possible to conclude that most of the existing approaches are tailored to work over a single data model since the DCOs are (or must be) specified in a language (e.g. SQL, SPARQL [W3C 2013a], SmartClean [Oliveira et al. 2009]) that is directly supported by the data management systems implementing such data model. Moreover, it was also observed that DCOs rely and reflects the data structure of the target repository preventing its applicability on other repositories about the same

domain but with a different structure and/or model. Thus, these approaches are only suitable for scenarios were all repositories share the same model and structure.
To overcome the described limitations, a novel data cleaning methodology was recently proposed [Almeida et al. 2015] and [Almeida et al. 2015a]. In summary, this approach grounds on two core ideas:

- the domain conceptualization is captured broadly and generically through an OWL ontology, which is called as the domain ontology;
- the structure and semantics of (concrete) DCOs is univocally described through a vocabulary (called DCOV) that is independent of any domain/application. In fact, this vocabulary is provided by the E-DQM ontology, which is an extended version of the DQM ontology proposed in [Almeida et al. 2015a].

Therefore, this approach comprehends a set of processes aiming (i) to establish mappings between the model/schema elements of a given repository and the elements of the domain ontology; and (ii) to rewrite the DCOs specified by the domain expert regarding the domain ontology and the DCOV according to the requirements of an existing data cleaning tool that is responsible for executing the DCOs on the target repository.
The work described in this paper is focus on this novel data cleaning methodology and, in particular, in its Data Cleaning Operation Rewriting Process (DCORP). Hence, we start with the context of this work. Then by clearly state the problem addressed by DCORP and identify its relevant dimensions (cf. section 3). Next, based on the dimensions previously identified, it is described four possible scenarios faced by the DCORP (cf. section 4). Further, in section 5, our proposal to deal with such scenarios is presented. Then, in section 6, our proposal is discussed regarding some relevant related work. At last, in Section 7, some conclusions are drawn together with some future work directions.

2. CONTEXT

This work occurs in the scope of the ontology-based data cleaning methodology proposed in [Almeida et al. 2015] and [Almeida et al. 2015a]. This methodology comprehends a Data Cleaning Operation Rewriting Process that only takes place assuming that:

- a domain expert has already specified (at a conceptual level) a set of DCOs for a given domain ontology; and
- the structure and semantics of such DCOs is described by the E-DQM ontology.

Next, these concepts are introduced and exemplified by setting up a simple running example scenario.
Consider that regarding a customers' domain, an organization is using the ontology depicted in Figure 1 (a), which comprehends the class customer (*o:Customer*) and properties for capturing customer´s name (*o:name*), address (*o:address*), tax number (*o:taxNumber*), the maximum discount allowed (*o:maxDiscount*) in percentage, its gender (*o:gender*) and the last time that customer' information was updated (*o:lastUpdate*). Moreover, consider that within this scenario, the domain expert aims to guarantee that:

- All customers have a name value (G_1);
- The tax number of each customer satisfies the syntax of Portuguese tax numbers (G_2).

Each guarantee may be verified by means of a detection DCO. The relevant excerpt of the E-DQM ontology that is required to specify such DCOs is depicted in Figure 1 (b).

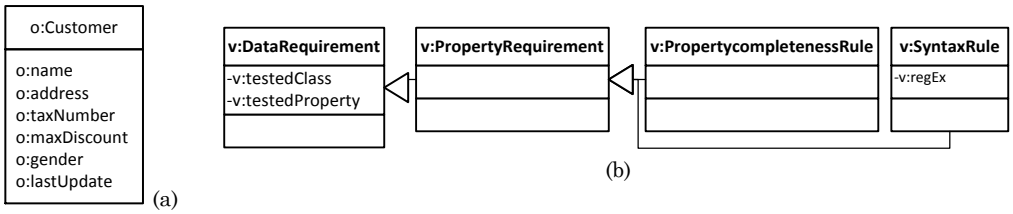

Fig. 1. Extract of a customers' domain ontology (a) and extract of the E-DQM ontology (b), both represented in UML.

This ontology captures and classifies different kinds of DCOs through a hierarchy of classes. Regarding detection DCOs only, the most generic class is *v:DataRequirement*. Its sub-classes (e.g. *v:PropertyRequirement*) serves as a logical arrangement capturing a set of characteristics shared by its leaves sub-classes (e.g. *v:SyntaxRule*). So, just the leaves classes are worth of being instantiated. Class *v:PropertyRequirement* represents DCOs acting on a single property (*v:testedProperty1*) of a given class (*v:testedClass*), while DCOs acting on at least two properties (*v:testedProperty1* to *v:testedPropertyN*) are captured by the *v:MultiPropertyRequirement* class. These kind of DCOs intend, for example, to check which instances of a class do not have a value on a given property (*v:PropertyCompletenessRule*), or, instead, to check if the value of such property is valid. The validity of a value may be expressed using a variety of approaches such as (i) specifying (in)valid ranges and/or (ii) regular expressions (*v:SyntaxRule*).
Table I illustrates the usage of the E-DQM ontology to specify the detection DCOs (DCO_1 and DCO_2) that allow to verify the previously described guarantees (G_1 and G_2 respectively).

Table I. DCOs specified in E-DQM on the domain ontology using the Turtle syntax

Detection Data Cleaning Operation	
DCO_1	***DCO_2***
:dco1 a v:PropertyCompletenessRule; v:testedClass o:Customer; v:testedProperty1 o:name.	:dco2 a v:SyntaxRule; v:testedClass o:Customer; v:testedProperty1 o:taxNumber; v:regex "^([1-3][0-9] | [45][0-9]{7}$)".

3. PROBLEM DEFINITION

The problem addressed by the Data Cleaning Operation Rewriting Process (DCORP) is graphically depicted in Figure 2.

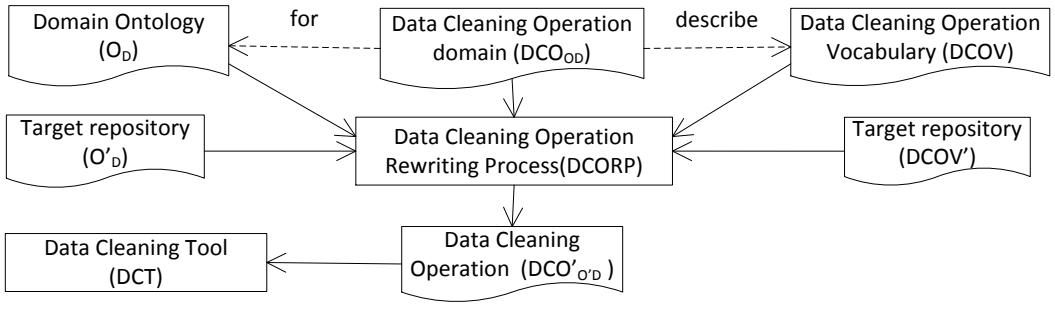

Fig. 2. Graphical representation of the DCORP addressed problem.

As that, besides the elements described in the previous section: (i) the domain ontology (O_D), (ii) the vocabulary (DCOV) used to describe DCO (i.e. E-DQM ontology) and (iii) the DCOs specified by the domain expert (DCO_{O_D}); the rewriting process (DCORP) has the following inputs:

- A data structure specification (O'_D) of the (target) repository on which one aims to execute the DCOs defined by the domain expert at a conceptual level (DCO_{O_D});
- A specification of the language and/or vocabulary ($DCOV'$) wherein the DCO_{O_D} should be rewritten. This vocabulary ($DCOV'$) is determined by the data cleaning tool (DCT) that is responsible to execute the DCOs on the target repository.

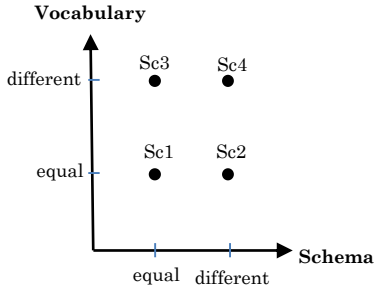

Fig. 3. Dimensions for DCO´s transformation.

The DCORP output is a set of DCOs ($DCO'_{O'_D}$) structurally and semantically described according to the $DCOV'$ and in respect with the data structure of the target repository (O'_D) such that it can be directly executed by the required DCT. Hence, the DCORP is formally seen as a function $DCORP(DCOV, O_D, DCO_{O_D}, DCOV',\ O'_D) \rightarrow DCO'_{O'_D}$.

4. PROBLEM ANALYSIS

According to the provided problem definition, one can say that the problem in hands implies to deal with variations regarding to two distinct dimensions:

- The data schema dimension, which captures variability concerned with the data structure of the domain ontology used to specify DCOs and the data structure in which such DCOs are expected to be executed;
- The vocabulary dimension, which captures variability concerned with the vocabulary or language in which DCOs are represented. Since the E-DQM is not a standard[1] most likely the existing data cleaning tools require that DCOs are specified in accordance with another vocabulary.

Thus, considering these two dimensions and for each dimension two possible values: one stating that there is no variation on the dimension (*equal*) and another stating the contrary (*different*), DCORP has to handle with four possible scenarios (cf. Figure 3). Each scenario (*Sc1* to *Sc4*) is briefly introduced along the following sub-sections.

[1] To the best of our knowledge, at the moment, there is no standard or recommendation that is widely accepted for representing the structure and semantics of data cleaning operations.

4.1 Scenario 1

Concerning to the first scenario (*Sc1*), there is no variation regarding either the data schema and the vocabulary dimensions (i.e. $O_D \equiv O'_D$ and $DCOV \equiv DCOV'$). Thus, in this scenario it is not necessary to rewrite the DCOs specified by the domain expert over the domain ontology since the target repository has the same schema of the domain ontology and the data cleaning tool responsible for the execution of such DCOs is able to understand and execute DCOs described through the same DCOV, which in this case is the E-DQM ontology. So, it can be concluded that $DCO_{O_D} \equiv DCO'_{O'_D}$. As that, the DCORP can be simply seen as a function $DCORP_{Sc1}(DCOV, O_D, DCO_{O_D}) \rightarrow DCO_{O_D}$.
Yet, it is worth to mention that in the scope of this work, this scenario is not likely to happen.

4.2 Scenario 2

Concerning to the second scenario (*Sc2*), the variation occurs on the data schema dimension only (i.e. $O_D \not\equiv O'_D$ and $DCOV \equiv DCOV'$). Accordingly, the DCORP can be seen as a simplified function $DCORP_{Sc2}(DCOV, O_D, DCO_{O_D}, O'_D) \rightarrow DCO_{O'_D}$.
Example 3.1 briefly describes a partial view of a data schema distinct of the one used in the running example (i.e. O_D). Now, considering that an organization intends to apply the DCOs represented on Table I to the relational repository presented in Example 3.1, a plausible output would be similar to the one illustrated in Table II (changes regarding Table I are emphasized with underlined text).
Example 3.1 (Schema of target repository). Consider that an organization has a relational database about customers whose structure comprehends a table (*db:Buyer*) used to record buyers information such as its name (*db:name*), its VAT number (*db:vatNumber*) and its billing address (*db:billAddress*) as depicted in Figure 4 (a).

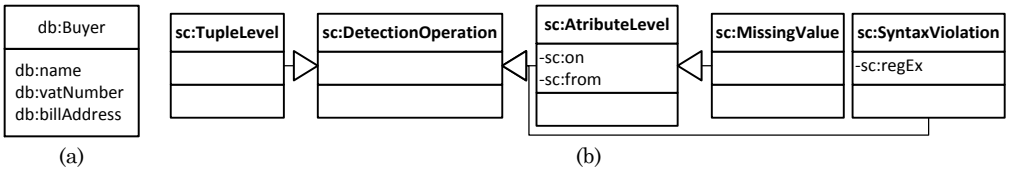

Fig. 4 Example of a (target) repository schema about customers (b) Excerpt of the SmartClean taxonomy represented in UML.

Table II. DCOs specified in E-DQM on the target repository schema using the Turtle syntax

Detection Data Cleaning Operation	
DCO₁	***DCO₂***
:dco1 a v:PropertyCompletenessRule; v:testedClass db:Buyer; v:testedProperty1 db:name.	:dco2 a v:SyntaxRule; v:testedClass db:Buyer; v:testedProperty1 db:vatNumber; v:regex "^([1-3][0-9] | [45][0-9]{7}$)".

4.1 Scenario 3

Concerning to the third scenario (*Sc3*), the variation occurs on the vocabulary dimension only (i.e. $O_D \equiv O'_D$ and $DCOV \not\equiv DCOV'$). Accordingly, the DCORP can be seen as a simplified function $DCORP_{Sc3}(DCOV, O_D, DCO_{O_D}, DCOV') \rightarrow DCO'_{O_D}$.
In order to illustrate this kind of variation, SmartClean [Oliveira et al. 2009] is briefly introduced as it is simultaneously a tool and a taxonomy tailored to handle DQP in

relational databases. This taxonomy is partially represented in Figure 4(b), in which, in light of our running example, are highlighted the most relevant elements.
Similarly, to E-DQM ontology, SmartClean taxonomy also captures and classifies different kinds of DCOs through a hierarchy of classes. Since it is target for relational databases, the adopted terminology is very closed to the one used on that field. On which concerns to detection DCOs, the root class is *sc:DetectOperation*. This class if further specialized regarding different levels of action such as (i) the attribute level (*sc:AttributeLevel*) and (ii) the tuple level (*sc:TupleLevel*); besides others. Focusing on the attribute level, there are several kinds of DCOS. In particular, it is highlighted DCOs checking for (i) missing (or null) values (*sc:MissingValue*) and (ii) a syntax violation (*sc:SyntaxViolation*) on an attribute (*sc:on*) of a given table or view (*sc:from*). The latter, also requires the specification of a regular expression (*sc:regEx*). At last, despite the use of this taxonomy, the SmartClean tool requires DCOs to be represented in plain text as illustrated in Example 3.2.

Example 3.2 (SmartClean DCO). A DCO aiming to detect which customers have no value on its name value (i.e. similar to DCO_1) using the data structure of the domain ontology (i.e. O_D) is represented in SmartClean through the following plain text: "DETECT MISSING-VALUE ON o:name FROM o:Customer OF o". Uppercase words derive directly from the taxonomy, while underline words refer to the schema elements used to instantiate the rule. The "OF" keyword and its value is a pointer to the repository on which DCOs are about to be executed.

4.2 Scenario 4

Concerning to the fourth scenario (*Sc4*), the variation occurs simultaneously on both dimensions (i.e. $O_D \not\equiv O'_D$ and $DCOV \not\equiv DCOV'$). This kind of variation occurs, for instance, by the need of rewriting DCO_1 and DCO_2 (cf. Table I) in accordance with (i) the data schema described in Example 3.1 and (ii) using the SmartClean taxonomy depicted in Figure 4b. A plausible result would be similar to the one illustrated in Table III. Thus, this represents the most complex scenario of the problem in hands. Yet, currently it is seen as the most probable scenario.

Table III. DCOs specified in SmartClean on the target repository schema.

Detection Data Cleaning Operation	
DCO₁	***DCO₂***
DETECT MISSING-VALUE ON db:name FROM db:Buyer OF db	DETECT SyntaxViolation ON db:vatNumber FROM db:Buyer WHERE (regex "^([1-3][0-9] | [45][0-9]{7}$)".) OF db

5. PROPOSAL

This section describes the conceptual approach devised to tackle the analyzed problem, namely *Sc1* to *Sc4*. The overall approach is graphically depicted in Figure 5 and grounds on the following mindset and assumptions.

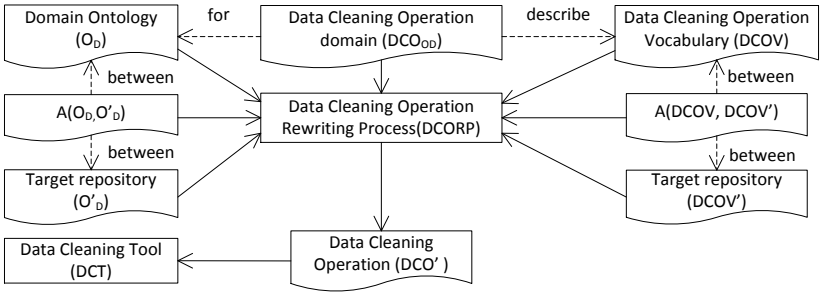

Fig. 5. Proposed conceptual approach.

Both the target repository schema (O'_D) and the vocabulary, on which the DCOs are required ($DCOV'$), are provided using the same formalism adopted to specify the domain ontology (O_D) and the E-DQM ontology ($DCOV$). In the context of this work, the adopted formalism is RDF/OWL [W3C 2004]. To fulfil this requirement, it might be necessary to make use of two external Lift and Normalization processes.

To deal with the (structural and semantic) heterogeneity between the two input schemas (O_D and O'_D), a set of correspondences (called Alignment) between the conceptual entities (e.g. classes, properties) of both schemas is assumed to be available ($A(O_D, O'_D)$). In this respect, it is worth to notice that it is expected that O_D and O'_D have, at least, some overlapping domain and, therefore, some correspondences exist. Yet, a correspondence establishes a relation (e.g. equivalence, subsumption, disjoint) between entities of a (source) schema and entities of another (target) schema. Table IV shows a possible alignment between the schemas depicted in Figures 1(a) and 4(a). Alignments may result from an external semi-automatic matching process.

Similarly, the heterogeneity between the input vocabularies ($DCOV'$ and $DCOV$) is also addressed assuming that an alignment ($A(DCOV, DCOV')$) between both vocabularies is available. Table V shows a possible alignment between the E-DQM ontology and the SmartClean extracts depicted in Figures 1(b) and 4(b) respectively.

Bearing in mind that the internal DCORP formalism is RDF/OWL which, in turn, is an abstract model, an optional input parameter (called *out_format*) is required in order to state the output (or serialization) format (e.g. Turtle [W3C 2013b], JSON-LD [W3C 2013c] plain text) in which the resulting DCOs ($DCO'_{O'_D}$) are desired (cf. Example 3.2).

Considering the exposed above, the DCOPR process consist on executing Algorithm 1, where:

- function $transform_{Sc2}(dco_{O_D}, A(O_D, O'_D))$ exploits the provided alignment (e.g. Table IV) to transform the inputted DCO (e.g. Table I) in another DCO according to the target schema (e.g. Table II);
- function $transform_{Sc3}(dco_{O'_D}, A(DCOV, DCOV'))$ exploits the provided alignment (e.g. Table V) to transform the inputted DCO (e.g. Table I) in another DCO according to the required vocabulary (e.g. Table VI);
- function $validate(dco'_{O'_D}, O'_D, DCOV')$ checks if the DCO resulting from previous transformation is valid. Notice that due to insufficient (or inexistent) correspondences a given transformation might not be possible;
- function $serialize(DCO'_{O'_D}, out\_format)$ serializes all the rewritten DCOs according to the specified output format.

The result of rewriting the DCOs represented in Table I in accordance with (i) the data schema described in Example 3.1; (ii) using the SmartClean taxonomy depicted in

Figure 4(b) and (iii) the alignment represented in Table IV and V is represented in Table VII (before serialization) and Table II (after serialization).

Table IV. Possible alignment between schemas adopted in Scenario 2

Entities		**Relation**
Source Schema (O_D)	**Target Schema (O'_D)**	
o:Customer	*db:Buyer*	Equivalence (≡)
o:name	*db:name*	Equivalence (≡)
o:taxNumber	*db:vatNumber*	Equivalence (≡)
o:Address	*db:billAddress*	Subsumption (≥)

Table V. Possible alignment between the provided extracts of E-DQM and SmartClean

Entities		**Relation**
E-DQM	**SmartClean**	
v:DataRequirement	*sc:DetectOperation*	*Equivalence (≡)*
v:PropertyRequirement	*sc:AttributeLevel*	*Equivalence (≡)*
v:PropertyCompletenessRule	*sc:MissingValue*	*Equivalence (≡)*
v:SyntaxRule	*sc:SyntaxViolation*	*Subsumption (≤)*
v:testedClass	*sc:from*	*Equivalence (≡)*
v:testedProperty1	*sc:on*	*Subsumption (≤)*
v:regex	*sc:regex*	*Equivalence(≡)*

ALGORITHM 1. DCORP Algorithm

Input: $DCOV, O_D, DCO_{O_D}, DCOV', O'_D, A(O_D, O'_D), A(DCOV, DCOV'), out\_format$.
Output: $DCO'_{O'_D}$.
$DCO'_{O'_D} \leftarrow \emptyset$;
for each $(dco_{O_D} \in DCO_{O_D})$
  if $(O_D \not\equiv O)$ **then** // Scenario 2 holds
    $dco_{O'_D} \leftarrow transform_{sc2}(dco_{O_D}, A(O_D, O'_D))$
  else
    $dco_{O'_D} \leftarrow dco_{O_D}$
  end
  if $(DCOV \not\equiv DCOV')$ **then** // Scenario 3 holds
    $dco'_{O'_D} \leftarrow transform_{sc3}(dco_{O'_D}, A(DCOV, DCOV'))$
  else
    $dco'_{O'_D} \leftarrow dco_{O'_D}$
  end
  $Valid \leftarrow validate(dco'_{O'_D}, O'_D, DCOV')$ // returns true or false
  if $Valid$ **then**
    $DCO'_{O'_D} \leftarrow \{DCO'_{O'_D}, dco'_{O'_D}\}$
  end
end
$\boldsymbol{serialize\left(DCO'_{O'_D}, out\_format\right)}$

Table VI. DCOs specified in SmartClean on the domain ontology using the Turtle syntax

Detection Data Cleaning Operation	
DCO₁	***DCO₂***
:dco1 a sc:MissingValue; sc:from o:Customer; sc:on o:name.	:dco2 a sc:SyntaxViolation; sc:from o:Customer; sc:on o:taxNumber; sc:regex "^([1-3][0-9]|[45][0-9]{7}$)".

Table VII. DCOs specified in SmartClean on the target repository schema using the Turtle syntax

Detection Data Cleaning Operation	
DCO₁	***DCO₂***
:dco1 a sc:MissingValue; sc:from db:Buyer; sc:on db:name.	:dco2 a sc:SyntaxViolation; sc:from db:Buyer; sc:on db:vatNumber; sc:regex "^([1-3][0-9] | [45][0-9]{7}$)".

6. RELATED WORK

In this section are presented approaches found in the literature that are related with this work. Almost all of them are concerned only with the differences that exist at the schema level, *i.e.*, just one of the dimensions considered in our work. At this level, the problem has deserved attention from the scientific community since a long time ago. Actually, the differences at the schema level among databases have always raised difficulties to the writing and execution of SQL queries. In [Wiederhold 1992] a mediator-based architecture was proposed to deal with the heterogeneities of distributed data sources. Its purpose is to provide an interface for accessing data which is able to support the process of their integration. Since then, several evolutions of this architecture have been proposed, as the peer to peer (P2P) architecture [Calvanese et al. 2006]. Still at the schema level, but more specifically in the data cleaning scope, the problem has also been studied by the research community. In [Wang et al. 2005]the authors propose the OntoClean framework which is a set of ontologies describing the domains represented by the classes and their attributes. Using the ontology based approach, it is possible to clean the data of not only syntactic errors but also some classes of semantic errors. In [Oliveira et al. 2006]the authors propose an approach that supports the interoperability of the data cleaning operations among different databases. That is achieved through an ontological level that supports the conceptual specification of the cleaning operations. The purpose of this abstraction level is to isolate the operations from the schema of the databases and support their reuse.

There are also approaches that tackle both dimensions, i.e., schema and vocabulary. The OntoDataClean system [Perez-Rey et al. 2006] deals with data integration at instance level and data preprocessing in the context of a process of knowledge discovery in databases. A preprocessing ontology was built to store the information about the required data transformations. Although the tool does not cover every possible data transformation, it suggests that ontologies are a suitable mechanism to improve quality in the various steps of the knowledge discovery process. This system is devoted only to the relational data model while our approach can handle different underlying data models. In [Fürber and Hepp 2011] the authors provide a conceptual model that allows the representation of data quality rules and other quality related knowledge using the Resource Description Framework (RDF) and the Web Ontology Language (OWL). Based on the proposed model, it is possible to monitor and assess the quality of data sources and to automate data cleaning tasks. The DQM ontology proposed by these authors was extended in the scope of our work (E-DQM, as previously mentioned) to cover additional data quality problems. In our work we also take advantage of the new/extended data cleaning vocabulary to support the rewriting of the operations to the language used by different data cleaning tools.

7. CONCLUSIONS AND FUTURE WORK

This work represents a step further in the methodology presented in [Almeida, Maio, Oliveira and João 2015] by proposing an approach for the rewriting process (DCORP). The approach takes a set of inputs which are related with the schema of the target repository and the vocabulary in which the DCOs are represented. For that, four scenarios that deal with the variations in these two dimensions were presented.

The process defined this way envisage to be the base to rewrite DCOs not only to the SmartClean tool, but to others tools or languages, like SQL, RIF, among others. To do that, there is always the need to specify the model of the language in which the DCOs are going to be specified and then executed. After that, there is the need to study the possibility of creating a template for the languages of the different tools according to similarities that may exist at a syntactical level. This is something which we intend to explore in the near future.

The process could be simplified if there was a standard language for the specification of DCOs. This would avoid the need to create different models for each of the vocabularies used by the data cleaning tools to detect the DQPs. This way, we would only have differences at the schema dimension.

Also as future work, we will continue the development of the SmartClean ontology and intend to refine the E-DQM in order to cover most of the data quality problems included in the taxonomy defined by Oliveira [Oliveira et al. 2005].

REFERENCES

Ricardo Almeida, Paulo Maio, Paulo Oliveira, and João Barroso. 2015a. An Ontology-based Methodology for Reusing Data Cleaning Knowledge: In SCITEPRESS - Science and and Technology Publications, 202–211. DOI:http://dx.doi.org/10.5220/0005596402020211

Ricardo Almeida, Paulo Maio, Paulo Oliveira, and Barroso João. 2015. Towards Reusing Data Cleaning Knowledge. In Alvaro Rocha, Ana Maria Correia, Sandra Costanzo, & Luis Paulo Reis, eds. *New Contributions in Information Systems and Technologies*. Advances in Intelligent Systems and Computing. Springer International Publishing, 143–150.

Luigi Atzori, Antonio Iera, and Giacomo Morabito. 2010. The internet of things: A survey. *Comput. Netw.* 54, 15 (2010), 2787–2805.

Grady Booch. 1993. *Object-Oriented Analysis and Design with Applications* 2 edition., Redwood City, Calif: Addison-Wesley Professional.

Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. 2006. Data management in peer-to-peer data integration systems. *Glob. Data Manag.* (2006), 177–201.

Edgar F. Codd. 1970. A relational model of data for large shared data banks. *Commun. ACM* 13, 6 (1970), 377–387.

Tamraparni Dasu, Gregg T. Vesonder, and Jon R. Wright. 2003. Data quality through knowledge engineering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 705–710.

Christian Fürber and Martin Hepp. 2011. Towards a vocabulary for data quality management in semantic web architectures. In *Proceedings of the 1st International Workshop on Linked Web Data Management*. ACM, 1–8.

Jing Han, E. Haihong, Guan Le, and Jian Du. 2011. Survey on NoSQL database. In *Pervasive computing and applications (ICPCA), 2011 6th international conference on*. IEEE, 363–366.

Magnus Knuth and Harald Sack. 2014. Data cleansing consolidation with PatchR. In *The Semantic Web: ESWC 2014 Satellite Events*. Springer, 231–235.

Diego Milano, Monica Scannapieco, and Tiziana Catarci. 2005. Using ontologies for xml data cleaning. In *On the Move to Meaningful Internet Systems 2005: OTM 2005 Workshops*. Springer, 562–571.

Paulo Oliveira, Maria de Fatima Rodrigues, and Pedro Rangel Henriques. 2006. An Ontology-Based Approach for Data Cleaning. In *ICIQ*. 307–320.

Paulo Oliveira, Fátima Rodrigues, and Pedro Henriques. 2009. SmartClean: An Incremental Data Cleaning Tool. In *Quality Software, 2009. QSIC'09. 9th Int. Conference on*. IEEE, 452–457.

Paulo Oliveira, Fátima Rodrigues, Pedro Henriques, and Helena Galhardas. 2005. A taxonomy of data quality problems. In *2nd Int. Workshop on Data and Information Quality*. 219–233.

David Perez-Rey, Alberto Anguita, and Jose Crespo. 2006. OntoDataClean: Ontology-Based Integration and Preprocessing of Distributed Data. In Nicos Maglaveras, Ioanna Chouvarda, Vassilis Koutkias, & Rüdiger Brause, eds. *Biological and Medical Data Analysis*. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 262–272.

Thomas C. Redman. 2001. *Data Quality: The Field Guide*, Digital Press.

S. Singh and N. Singh. 2012. Big Data analytics. In *2012 International Conference on Communication, Information Computing Technology (ICCICT)*. 1–4. DOI:http://dx.doi.org/10.1109/ICCICT.2012.6398180

Chris Snijders, Uwe Matzat, and Ulf-Dietrich Reips. 2012. Big data: Big gaps of knowledge in the field of internet science. *Int. J. Internet Sci.* 7, 1 (2012), 1–5.

W3C. 2004. OWL Web Ontology Language Reference. (2004). Retrieved April 12, 2016 from https://www.w3.org/TR/owl-ref/

W3C. 2013a. SPARQL 1.1 Query Language. (2013). Retrieved April 11, 2016 from https://www.w3.org/TR/sparql11-query/

W3C. 2013b. Turtle. (2013). Retrieved April 11, 2016 from https://www.w3.org/TR/2013/CR-turtle-20130219/

W3C. 2013c. JSON-LD 1.0. (2013). Retrieved April 11, 2016 from https://www.w3.org/TR/2013/PR-json-ld-20131105/

Xin Wang, Howard John Hamilton, and Yashu Bither. 2005. *An ontology-based approach to data cleaning*, Regina: Department of Computer Science, University of Regina.

Melanie Weis and Ioana Manolescu. 2007. Declarative XML data cleaning with XClean. In *Advanced Information Systems Engineering*. Springer, 96–110.

Gio Wiederhold. 1992. Mediators in the architecture of future information systems. *Computer* 25, 3 (1992), 38–49.

Received Month YYYY; revised Month YYYY; accepted Month YYYY