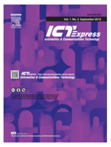
Available online at www.sciencedirect.com
ScienceDirect
ICT Express 3 (2017) 57–61
www.elsevier.com/locate/icte


A standards-based ontology and support for Big Data Analytics in the insurance industry☆,☆☆,★

Dimitrios A. Koutsomitropoulos*, Aikaterini K. Kalou

Computer Engineering and Informatics Dpt., HPCLab, University of Patras, Patras, Greece

Received 16 March 2017; received in revised form 14 May 2017; accepted 23 May 2017
Available online 3 June 2017

Abstract

Standardization efforts have led to the emergence of conceptual models in the insurance industry. Simultaneously, the proliferation of digital information poses new challenges for the efficient management and analysis of available data. Based on the property and casualty data model, we propose an OWL ontology to represent insurance processes and to map large data volumes collected in traditional data stores. By the virtue of reasoning, we demonstrate a set of semantic queries using the ontology vocabulary that can simplify analytics and deduce implicit facts from these data. We compare this mapping approach to data in native RDF format, as in a triple store. As proof-of-concept, we use a large anonymized dataset for car policies from an actual insurance company.

Keywords: Insurance industry; Big data; OBDA; Triple stores; Reasoning

1. Introduction

The insurance sector is well-known for its need for accurate knowledge-based decisions and high data availability. The efficient and meaningful manipulation of growing data volumes means that not only should insurance companies keep track of them, but also devise intelligent strategies for their analysis. Semantic technologies (Ontologies, Linked Data) can have considerable repercussions in addressing Big Data issues for insurance.

An ontology defines a common set of terms to represent the basic concepts in a domain and the relationships between them. With the power of reasoning, new facts, which are not explicitly expressed in an ontology, can be derived. Therefore, additional knowledge for further analysis can be provided, including more accurate risk identification and assessment. This kind of semantic analytics can help companies alleviate long-standing problems, as well as improve standard business processes. Sectors such as customized insurance policies, fraud detection, and marketing, benefit from the application of semantic analytics methods over Big Data [1].

In this work, we first propose an ontology implementation[1] of the Property & Casualty (P&C) data model that has emerged as a conceptual representation standard among insurance stakeholders. Next, we develop an infrastructure that allows for the association of the ontology with raw data in real time and for inferring knowledge without sacrificing considerable performance. For our experiments, we use a large dataset from an existing vehicle insurance company. We show that it is possible to apply semantic analytics to these data and deduce meaningful facts, based on a series of queries to the model that are processed by a reasoner. Queries are performed on-the-fly, with data remaining in a relational database. For the purposes

* Corresponding author.
E-mail addresses: kotsomit@ceid.upatras.gr (D.A. Koutsomitropoulos), kaloukat@hpclab.ceid.upatras.gr (A.K. Kalou).
Peer review under responsibility of The Korean Institute of Communications Information Sciences.
☆ A preliminary version of this paper has been presented at the 2nd INNS Conference on Big Data (INNS Big Data 2016).
☆☆ This paper is part of a special section titled "Special Issue on Patents, Standardization and Open Problems in ICT Practices".
★ This paper has been handled by Dong-Soo Han.

[1] Available at: https://goo.gl/XVZOXk.

http://dx.doi.org/10.1016/j.icte.2017.05.007

of evaluation, we compare this "virtual-graph" approach against native RDF data in a triple store, and utilize inference in both cases.

The rest of this paper is organized as follows: In Section 2, we briefly review the use of semantic technologies over Big Data and the approach of OBDA (Ontology Based Data Access). In Section 3, we present the P&C ontology and explain its development and usage. Section 4 outlines our methodology for data access and manipulation. Section 5 demonstrates intelligent queries performed over insurance data and discusses our comparison results. Finally, Section 6 summarizes our conclusions and future work.

2. Related work

Recent surveys indicate that P&C insurance companies are soon going to be reformed into being data-driven by leveraging the opportunities of Big Data. In addition, semantic technologies are gradually becoming an asset to the insurance industry [2].

NoSQL, Yarn, Hadoop, MapReduce, and HDFS are among the most well-known technologies for handling Big Data in a cost-effective manner [3]. Moreover, semantic triple stores are continually evolving towards the vision of exploring large datasets. For example, Oracle Spatial and Graph with Oracle Database 12c, AllegroGraph, Stardog, and OpenLink Virtuoso v6.1 have expanded their scalability and performance features to meet these requirements [4].

Efficient data integration is of key importance for Big Data, as it often demands the knowledge of schema and the format of each data source. Within OBDA systems [5], an ontology is used to expose data in a conceptual manner, by abstracting away from the schema-level details of the underlying data. The connection between data and ontology is performed via mappings. The combination of ontology and mappings allows automatic translation of queries posed over the ontology into data-level queries that can be executed by the specific underlying DBMS. Ontop, Mastro, Stardog, and D2R servers are among the most popular OBDA systems.

This "semantic overlay" not only enables reasoning-based retrieval of new facts, but also simplifies access to and insights on data. In the insurance sector, traditional access to data often requires complex SQL queries and syntax that are hard to comprehend and conceive for non-database experts. Further, it involves convoluted schemas and legacy warehouses, spanning over multiple relational tables, and often overlapping. The use of SPARQL [6], combined with the appropriate ontology, can assist in the formulation and execution of business requirements posed by queries [7].

3. Towards an insurance ontology

A starting point for the semantic formalization of the business functions and processes of the insurance sector is offered by the P&C data model, developed by the Object Management Group [8]. Its major components include *entities*, *attributes*, and *relationships*. An *Entity* represents a person, organization, place, object, or concept of interest to the enterprise. A *Relationship* is a verb phrase describing that which is always established between a *Parent* and a *Child* entity. *Attributes* are usually defined within an entity and are considered as properties or descriptors for this entity.

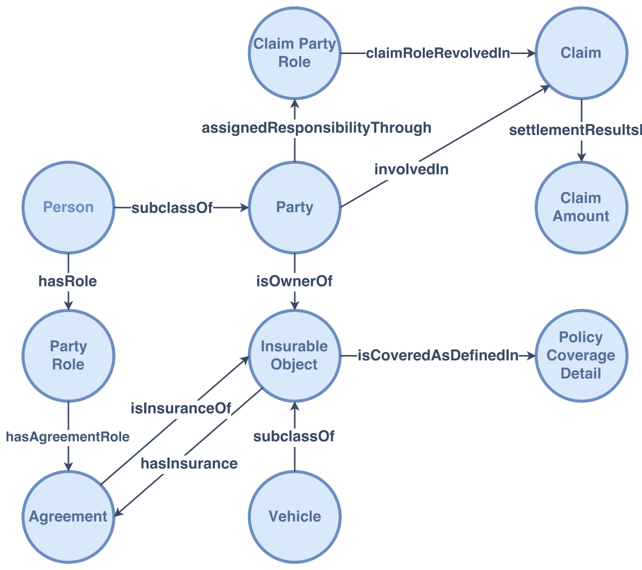


Fig. 1. Class and property relationships in the P&C.

To build an OWL (Ontology Web Language) ontology for P&C, we start by simply correlating the logical model's entities to OWL classes, relationships to object properties and attributes to data properties [9]. Next, "subtype" relationships are used to construct the class hierarchy [10]. Parent and child entities form a property's domain and range, respectively. In addition to these constructs, we specified certain additional logical axioms available in OWL and useful for inferencing: for example, a property with a max cardinality of one is functional, and *hasOwner* and *isOwnerOf* are inverses. Fig. 1 depicts part of the class hierarchy of the P&C ontology and the object properties relating the instances of these classes. A simple conceptualization example in natural language involving ontology terms is as follows (class instances are in italics and property names are underlined):

A *Vehicle_1* has insurance in the form of *Agreement_1* and is covered as defined in *PolicyCoverageDetail_1*. After an accident involving material damages, *Person_1*, the owner of *Vehicle_1*, has filed and is involved in *Claim_1*. A fraud assessment on this claim turns out negative; thus, the claim comes to a settlement that results in a specific amount, *ClaimAmount_1*, reimbursed by the insurance company. Therefore, *Claim_1* can now be classified as *AlreadySettled*.

4. Big data access and manipulation

We considered large volumes of offline data originating from a well-known vehicle insurance company. The data were SQL dumps, each corresponding to a separate relational table. For a trivial mapping, a single column of each tuple forms a separate RDF triple (Table 1).

Table 1
Insurance dataset metrics.

Relational schema	Tables	5
	Columns per table	31 (on average)
	Tuples	0.5M per table
Ontology	Classes	95
	Object properties	26
	Datatype properties	17
Triples	Trivial mapping count	77,000,000
	Actual count reported	33,673,002

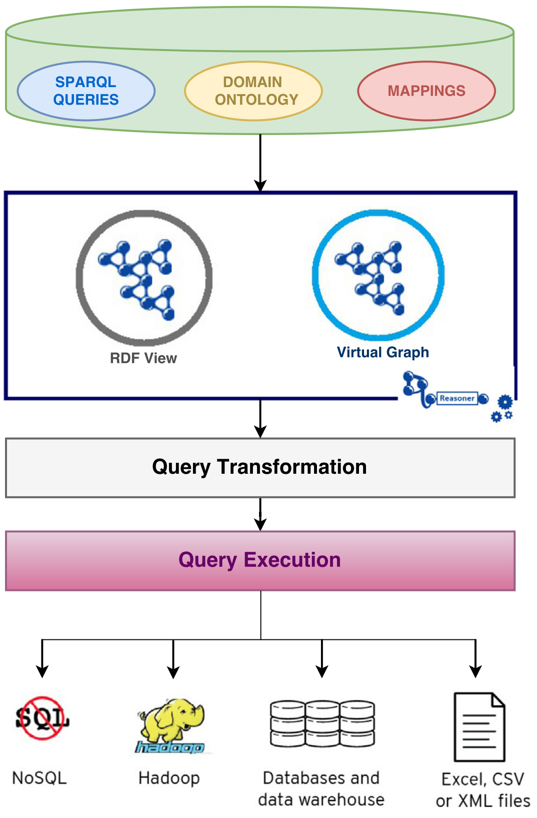


Fig. 2. Query evaluation workflow.

For our OBDA infrastructure, we selected Ontop [11] owing to its ease of use, intuitive mapping support, and high performance capabilities. In addition, Ontop supports reasoning at the OWL 2 QL level [12], which is a lightweight reasoning profile, but sufficiently expressive to support inferences on very large volumes of instance data. The data is accessed in the form of *virtual graphs* based on the P&C OWL ontology, and a set of mappings pointing from the relational data to ontology triples (see Table 2). Queries can then be performed using SPARQL, by considering the mappings already defined and the outcomes of the reasoning process on the ontology (Fig. 2).

This occasional access to triples (i.e., materialization of only those parts of the graph deemed relevant to the particular query) can be contrasted to full graph materialization. In the latter case, all data are mapped to the ontology in advance and the whole set of triples is readily available, e.g., within a triple store. We identify three main differences between the two approaches.

Performance: in principle, the ability to decompose a SPARQL query into a set of SQL queries signifies a reasoning algorithm with computational complexity similar to that of relational databases. OWL reasoners are known for high worst-case complexity bounds [12]. However, most contemporary reasoners implement a series of optimizations and exhibit a pay-as-you-go behavior, or rely on scalable rule-based algorithms.

Expressiveness level of inferences: the Ontop internal reasoner is limited to the OWL 2 QL subset, which for example, does not allow for class disjointness, transitivity in properties, or class membership based on property restrictions. It is exactly owing to this limitation that queries can be rewritten into SQL queries without loss, and OWL 2 QL was designed with this in mind. On the other hand, access to the full graph facilitates the employment of more expressive reasoners, thus performing a more insightful and knowledge-intensive analysis.

Support for streaming data: SPARQL 1.1 adds support for updates to graphs and this is implemented in most triple stores. Therefore, it is possible to update the knowledge base with data arriving at any rate. In this case, data that used to be stored in a relational database needs to be transmitted directly to the triple store, i.e., it must already be triplified or imported at certain intervals.

5. Semantic analytics

5.1. *Intelligent queries*

We present four sample SPARQL queries over the insurance dataset, most of which yield results due to some form of reasoning. They actually represent "query families", each exploiting some unique characteristic of the ontology allowed in the OWL 2 QL profile, such as inheritance, hierarchy, inverse property, and existential restriction. Queries can be considered in the context of the usage scenario outlined in Section 3 and Fig. 1.

```
SELECT ?x WHERE { ?x a :InsurableObject.}  (Q1)
```

In Q1, although we have not designed a mapping rule defining instances of the *InsurableObject* class, the instances of *Vehicle* are automatically classified as such, as *Vehicle* is defined as a subclass of *InsurableObject* in the ontology.

```
SELECT ?y ?z WHERE {
      :367788 :isOwnerOf ?y.
      ?z :isInsuranceOf ?y.}  (Q2)
```

Q2 retrieves the insurance policies for all vehicles that are owned by a specific *Party*. Note that we can use the *isInsuranceOf* property in the query, instead of *hasInsurance* as specified by mapping #1 in Table 2, because they are defined as inverses in the ontology.

```
SELECT ?v ?y WHERE {
      ?v :involvedIn ?y.
      ?y a :AlreadySettled.}  (Q3)
```

With mapping #3, we relate a *Vehicle* to a *Claim* and the *Claim* to its resulting settlement *ClaimAmount*. Q3 discovers

Table 2
A partial set of mappings used for the semantic modeling of insurance data.

#	Source (SQL Query)	Target (Triples Template)
1	SELECT plate, contract FROM InsuredItemVehicle;	:{plate} a: Vehicle; :hasInsurance :{contract}. :{contract} a :Policy
2	SELECT customerCode as c, iv.plate FROM InsuredItemCustomer as ic, InsuredItemVehicle as iv WHERE ic.contract = iv.contract;	:{c} a :Person; :isOwnerOf :{plate}.
3	SELECT policyNumer as pol, ClaimNumber as r, totalPayAmount as amnt, iv.contract, plate FROM Claims, InsuredItemVehicle as iv where pol = iv.contract;	:Amount_{pol}_{r} :hasAmount {amnt}^^xsd:decimal. :Claim_{pol}_{r} a :Claim; :settlementResultsIn :Amount_{pol}_{r}. :{plate}:involvedIn :Claim_{pol}_{r}

Table 3
Query performance results—execution time (s).

	Q1		Q2		Q3 (Q5)		Q4	
	Time	#results	Time	#results	Time	#results	Time	#results
Virtual graph	7.1	543398	9.3	72	6.0	85183	–	–
Triple store	3.9	543398	0.1	72	3.5	85183	22.3	89855
No reasoning	0.01	0	0.01	0	1.2	85183	6.4	89855

vehicles *involvedIn Claims* that have already been settled with a *ClaimAmount* through *settlementResultsIn*. This is possible using the auxiliary class *AlreadySettled*, specified as a superclass of an existential restriction on the *settlementResultsIn* property.

```
SELECT DISTINCT ?x z(SUM(?z) AS ?total)
WHERE { ?x :isOwnerOf ?k. ?k :involvedIn ?s.
?s :settlementResultsIn ?y. ?y :hasAmount ?z.}
GROUP BY ?x ORDER BY DESC (?total)  (Q4)
```

Q4 displays the sum amount reimbursed by the company to its customers over all their settled claims, i.e., it reveals the most 'expensive' customers.

5.2. *Results and discussion*

All experiments were conducted on standard commodity hardware. A 4-core laptop CPU was used and Java was assigned 4 GB for heap memory. To put data into the triple store (Fuseki), we used Ontop's *materialize* command that runs the mappings and produces all possible triples. Prior to the actual computation of the performance for both approaches (Table 3), we applied standard benchmarking techniques, such as clearing the cache and performing warm-up queries [13].

As expected, query evaluation without enabling reasoning in the triple store is considerably faster, but does not discover implied facts. We observe that the triple store outperforms OBDA by almost a factor of two. A notable exception is Q2, which involves only a single instance in the graph pattern; therefore, it is much faster owing to inference materialization performed in advance by the reasoner. For the virtual graph, all queries fall approximately within the same order of magnitude because a live SQL translation was performed.

Q3 involves inference, which is not supported by the OWL entailment regime implemented in our example triple store. In fact, Fuseki employs an incremental set of rule-based reasoners ranging from full RDFS support to OWL, minus certain constructs [14]. Thus, their expressivity is very close to, if not broader than, that of OWL 2 QL, given that it corresponds to the intersection of RDFS and OWL 2 DL. Of the three available OWL reasoners, only the smallest one ('OWLMicro') could feasibly ran the queries without reaching the available memory upper limit (8 GB). However, we can rewrite this query into:

```
SELECT ?v ?y WHERE {
    ?v :involvedIn ?y.
    ?y :settlementResultsIn ?z.}  (Q5)
```

which yields the same results and shows that, in cases such as these, existential quantification could be reduced to syntactic substitution [15].

Q4 uses the new, SQL-like aggregation functions of SPARQL 1.1 and cannot run with Ontop. However, Fuseki, along with most contemporary triple stores, can handle such queries successfully.

6. Conclusions and future work

The evolvement of industry data requirements in the Big Data era has caused semantic technology research and vendors drive to meet their expectations, by improving query access, performance, and inference added-value at a scale. An insurance ontology can play the role of a common, standardized vocabulary that helps communication and knowledge exchange between insurance partners, in the form of Linked Data. By taking advantage of the P&C ontology, we have shown that it is possible to simplify query formulation and execution and to enable reasoning-based queries that are not straightforward, or even possible, with relational databases.

However, there are justified performance considerations. We have demonstrated a couple of approaches to access data in triple form that seem to cope with this problem, each with its own tradeoffs. A possible improvement for both approaches is the consideration of performance with real-time data chunks, for example, through SPARQL updates. A further improvement is to investigate the possibility of reasoning with streaming data; a more extensive evaluation would be necessary in this case, since continuous changes in the datasets may affect performance results. Despite which approach is actually followed, we have shown evidence that industries facing ever-increasing datasets can make meaningful use out of them, and use semantic analytics as an advantage.

References

[1] B. Marr, How big data is changing insurance forever. Technical article, Forbes, 2015.

[2] W.T. Watson, How are life insurers planning to use big data and predictive analytics? 2016. Available at http://blog.willis.com/2016/12/how-are-life-insurers-planning-to-use-big-data-and-predictive-analytics.

[3] G. Press, Top 10 hot big data technologies. Technical article. Forbes, 2016.

[4] F. Michel, C. Faron-Zucker, J. Montagnat, A Mapping-based Method to Query MongoDB Documents with SPARQL, in: Proc. of the 27th International Conference on Database and Expert Systems Applications DEXA 2016, Porto, Portugal, 2016.

[5] A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini, R. Rosati, Linking data to ontologies, J. Data Semant. (2008) 133–173.

[6] S. Harris, A. Seaborne, (eds.). SPARQL 1.1 Query Language. W3C Recommendation, 2013.

[7] E. Kharlamov, D. Hovland, E. Jiménez-Ruiz, D. Lanti, H. Lie, Pinkel, et al., Ontology based access to exploration data at Statoil, in: Proc. of the 14th International Semantic Web Conference, Springer, 2015, pp. 93–112.

[8] W. Jenkins, R. Molnar, B. Wallman, T. Ford, Property and Casualty Data Model Specification. OMG, 2011.

[9] S. Soares, IBM InfoSphere: A Platform for Big Data Governance and Process Data Governance. MC Press Online, LLC, 2013.

[10] A.K. Kalou, D.A. Koutsomitropoulos, Linking data in the insurance sector: A case study, in: Proc. of the 10th Int. Conference on Artificial Intelligence Applications and Innovations, AIAI 2014, Springer, 2014, pp. 320–329.

[11] D. Calvanese, B. Cogrel, S. Komla-Ebri, R. Kontchakov, D. Lanti, M. Rezk, M. Rodriguez-Muro, G. Xiao, Ontop: Answering sparql queries over relational databases, Semant. Web 8 (3) (2017) 471–487.

[12] B. Motik, B.C. Grau, I. Horrocks, Z. Wu, A. Fokoue, C. Lutz, OWL 2 web ontology language profiles. W3C recommendation, 2012.

[13] S. Bail, S. Alkiviadous, B. Parsia, D. Workman, M. van Harmelen, R.S. Goncalves, C. Garilao, FishMark: A linked data application benchmark, in: Proc. of the Joint Workshop on Scalable and High-Performance Semantic Web Systems, SSWS+HPCSW 2012, Vol. 943, pp. 1–15, CEUR, ceur-ws.org, 2012.

[14] Apache Jena, Reasoners and rule engines: Jena inference support. Available at https://jena.apache.org/documentation/inference/.

[15] C. Corona, M. Ruzzi, D.F. Savo, Filling the Gap between OWL 2 QL and QuOnto: ROWLKit, in: Proc. of Description Logics 2009, CEUR, vol. 477, 2009. ceur-ws.org.