Semantic Big Data for Tax Assessment *

Stefano Bortoli
University of Trento
Dept. of Engineering and Information Science
Via Sommarive 14, Trento, IT
bortoli@disi.unitn.it

Paolo Bouquet
University of Trento
Dept. of Engineering and Information Science
Via Sommarive 14, Trento, IT
bouquet@disi.unitn.it

Flavio Pompermaier
Okkam SRL
Via Segantini 23, Trento, IT
pompermaier@okkam.it

Andrea Molinari
LUT - School of Industrial Engineering and Management
FI-53851 Lappeenranta, FI
andrea.molinari@unitn.it

ABSTRACT

Semantic Big Data is about the creation of new applications exploiting the richness and flexibility of declarative semantics combined with scalable and highly distributed data management systems. In this work, we present an application scenario in which a domain ontology, Open Refine and the Okkam Entity Name System enable a frictionless and scalable data integration process leading to a knowledge base for tax assessment. Further, we introduce the concept of *Entiton* as a flexible and efficient data model suitable for large scale data inference and analytic tasks. We successfully tested our data processing pipeline on a real world dataset, supporting ACI Informatica in the investigation for Vehicle Excise Duty (VED) evasion in Aosta Valley region (Italy). Besides useful business intelligence indicators, we implemented a distributed temporal inference engine to unveil VED evasion and circulation ban violations. The results of the integration are presented to the tax agents in a powerful Siren Solution KiBi dashboard, enabling seamless data exploration and business intelligence.

CCS Concepts

•**Information systems** → **Extraction, transformation and loading; Entity resolution;** *Data scans; Data cleaning; Spatial-temporal systems; Expert systems; Data analytics;* •**Computing methodologies** → *Parallel algorithms; Temporal reasoning;* Ontology engineering;

Keywords

Semantic Big Data; Inference; Tax assessment; Entity Name System

1. INTRODUCTION

The combination of semantic technology with big data tools opens new perspectives over the traditional data management systems. Pushing the representation of the semantic of data down to the level of data items (i.e. RDF Statements) gives a high level of flexibility in the definition of persistence and data representation models, compared to traditional relational databases. However, the redundant information increases the disk space required to store and handle the data, and requires the definition of novel data lifecycle processes. In this paper we show a real-world use case where the combination of semantic and big data technologies can provide great benefits without requiring expensive equipments. In particular, we describe a use case where these technologies have been used to define a Semantic ETL to support effectively and efficiently tax assessment activities to fight Vehicle Excise Duty (VED) evasion in the Aosta Valley region (Italy). The tools involved in the process are: 1) a **domain ontology** modeling concepts and relations among them; 2) an enhanced and extended version of **Open Refine**[1] as a data cleaning and manipulation tool [16]; 3) an instance of **Entity Name System** ([8]) supporting entity reconciliation and persistent identification across data sources; an **entity-centric data model** (a.k.a. *Entiton*) to cluster statements around entity identifiers; a combination of big data tools supporting efficient data storage, serialization and analysis (Apache Parquet[2], Apache Thrift[3] and Apache Flink[4]); and finally a data intelligence tool such as Siren Solution KiBi[5] to create dashboards and support seamless data exploration. The main objectives of the process are to integrate data coming from different sources, usually stored in relational databases, to produce an integrated knowledge base of *Entitons* that can be manipulated and analyzed to

*This work has been partially funded by the Autonomous Province of Trento (Legge 6/1999, DD n. 251) under the project SICRaS

SBD'16, July 01 2016, San Francisco, CA, USA

ISBN 978-1-4503-4299-5/16/07…$15.00

DOI: http://dx.doi.org/10.1145/2928294.2928297

[1] http://openrefine.org
[2] https://parquet.apache.org/
[3] https://thrift.apache.org/
[4] https://flink.apache.org/
[5] http://siren.solutions/kibi/

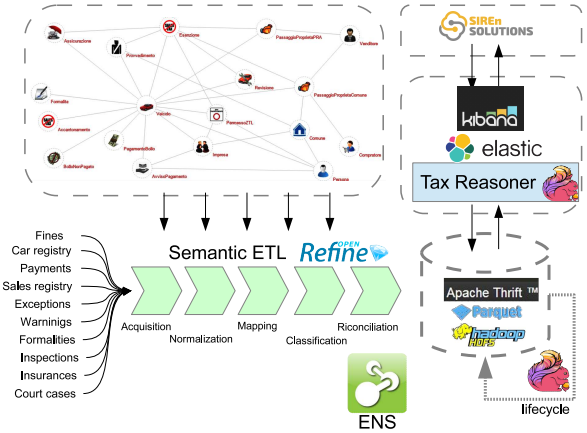

Figure 1: Semantic ETL for VED Assessment

infer information, and ultimately feed end-user application databases and indexes. The reminder of the paper is structured as follows: Section 2 describes the Okkam Semantic ETL; in Section 3 we briefly analyze the state of the art; in Section 4 describes the *Entiton* data model; Section 5 describes the analyzed use case and the involved datasets; Section 6 describes the implemented Tax Reasoner and the results; and finally we present some conclusions.

2. THE SEMANTIC ETL

In Figure 1, a graphical representation of the ETL is presented. The first step of the ETL process, inspired by the work described in [1], is the selection of the data sources including Open Data (e.g. municipal address registry and GeoNames[6]), purchased data (e.g. the list of all enterprises of the region), and dumps of relational databases. Among others, the sources considered include: the regional car registry (PRA[7]), fines, permissions, exemptions, vehicle inspection, and payments. The data collected usually consist of large CSV files provided by ACI Informatica, the IT company supporting ACI (Automobile Club Italia). These files are usually split per year, and include a wide range of attributes both related to properties of the considered entities (car, people, companies, etc.) and about events involving them (payments, fines, inspections, restrictions, exemptions, etc.). The main problem with these types of files, besides expected presence of errors and typos of real world data [11], are the implicit and often missing information. Furthermore, events are managed as punctual entities, defining just the date of the happening associated with some specific code in the database. To manage all these aspects we enhanced Open Refine, which allows for cleaning, transforming, mapping the data to the defined ontology[8], and crossing tables with related decoding tables. Once data are cleaned and normalized (e.g. street addresses mentioned in the data sources are matched towards the official municipality registry), we proceed to *okkamize* relevant entities according to the Okkam Conceptual Model [7]. Practically, we select a subset of columns of the source providing sufficient identification criteria for the considered type of entity, and submit queries to a purposely configured instance of the Entity Name System.

Currently, the ENS embeds the Fingerprint Matching Module, which implements a knowledge-based approach to the open entity matching problem [4]. If a query finds a matching entity (the ENS returns a URI and positive matching decision) than this is used to identify the target entity. In case of no positive matching candidates, a new persistent ID (a IRI) can be created for the entity in the ENS, enabling its reuse in the following iterations. If matching candidates are returned, but the ENS does not return a positive matching decision, it is possible to apply a set of matching rules that can be applied locally within Open Refine. Namely, there may be local identifiers that cannot be interpreted as globally inverse functional and diachronic attributes in a global context [5], but allow for reliable matching decision in the local dataset context. Hence, these properties (usually database identifiers), can be used to draw reliable matching decisions. In a similar way, it is possible to define also negative rules, reducing the possible effects of underspecified queries and filtering matching candidates which should not be considered by the operator managing the data source.

Once the cycles of interaction with the ENS are complete and the largest possible number of entities is *okkamized* (i.e. assigned a globally unique persistent identifier), data are exported through the RDF Extension[9] of Open Refine as a NQuad RDF[10] file. Crucial aspect of the process defined is the order of processing of the selected sources. For the moment, we rely on domain knowledge to consider reliability and richness of the different sources, and choose the most reliable and complete first, this way increasing the chances of a positive reconciliation.

The next step consists in processing the large NQuad RDF files to produce the *Entiton* data structure (described in details in section 4). An *Entiton* can be seens as an RDF rendering of the snowflake database model[2], where all statements about an entity are clustered around its identifier. Because we reconciled the identity of entities mentioned in different sources relying on the Entity Name System identifiers, we can now gather them and integrate them easily relying on a distributed process implemented with Apache Flink. The generated *Entiton* dataset, representing a first version of the knowledge base, are serialized using Apache Thrift, and stored using Apache Parquet. The combination of Apache Thrift+Parquet allows for very efficient scans applying the push-down filter technique, although penalizing random access. Experimental verification showed that loading 1 billion triples generated using BSBM [3] into the cluster described in Section 7 took 1h23', which is far below the best loading time described in [10]. Therefore, we accept the random access penalization as we consider the *Entiton* knowledge base as a primary data source used to feed the databases of end-user applications. Currently, we are interested in fast data scans to improve the efficiency of analysis inference processes to be executed in batch. At the end of the day, our customers want to use a purposely defined user interface that does not require knowing any query language.

The next step in the pipeline is the execution of reasoning tasks which materialize implicit knowledge to support

[6]http://www.geonames.org/
[7]Pubblico Registro Automobilistico, i.e. the national registry for vehicles
[8]http://models.okkam.org/tax/aci.owl
[9]http://refine.deri.ie/
[10]https://www.w3.org/TR/n-quads/

tax assessment. The details of the reasoning are presented in Section 6, after having explicitly clarified the use case scenario in Section 5. Relying on a domain ontology to harmonize the semantics of attributes and the ENS to reconcile entity identifiers heavily reduces the complexity of the inference process compared to traditional data warehouse solutions. When the inference process is complete and new knowledge is inferred, a set of administrative routines is executed to load and transform part of the knowledge base to feed applications databases and indexes. In our case, we process the *Entiton* data pool to generate a set of ElasticSearch[11] indexes used by the Siren Solution KiBi[12] plugin for Kibana[13]. The result is an adaptive dashboard that allows for a seamless exploratory analysis of the data where different visual elements update according to the current selected element (see Figure 2). Further, thanks to the inference described in Section 6 we can materialize information elements that are useful for the decision makers and for tax assessment in the car sector.

3. STATE OF THE ART

Calvanese et al. in [9] presented the MASTRO system, which allows for ontology-based data access to federated databases. The system exploits full reasoning tools, and the power of ontology navigation, keeping the data in the original sources. This has the advantage of working always with fresh data, but requires the definition of a large set of rigid mappings between the ontology and the structure of the databases. Furthermore, this does not solve the problem of reconciling the identity of entities mentioned in the federation of databases, solving mostly the problem of schema heterogeneity. In a sense, the MASTRO system offers a middleware schema-driven solution that is complementary to the one presented in this work. In fact, we focus is on entities and we apply a flexible approach to semantic harmonization (a.k.a schema matching) relying on contextual mappings [6] interpreted as analogies [14]. In [13], the authors propose the adoption of ontologies as a way to model dimensionality of data in data warehouse solutions. The ontology and constraints aim to capture the complex dimensions of an application domain to define OLAP data cubes for business intelligence. We can see this as a narrower alternative of the Mastro System, with similar advantages and suffering of the same issues. In [12] is presented an example of how ontology-based data access can be applied in a large enterprise such as Siemens. In particular, they introduce the Optique platform result of the FP7 EU funded project[14]. The Optique platform provides a semantic data integration platform, with a specific focus on supporting efficient end-user access to the data. However, Optique is based on relational database, and focuses on query translation and optimization, rather than enabling data enrichment and exploration. Optique supports domain experts in querying a large and complex knowledge base in an efficient way, whereas our approach has the objective of processing data to infer and materialize knowledge to satisfy the tax investigation requirements and enable navigation of the data space relying on responsive graphs and dashboard built on the ElasticSearch indexes. In [15] researchers explore the combination of big data and semantic technology in the medical domain, defining a pipeline that is similar to the one proposed in this work. However, the authors rely on SWRL rules and other description logic tools to infer information from data extracted from different sources which hits inherent complexity and therefore limitations in scalability compared with the analysis based on Apache Flink proposed in this work. Further, the authors outline a set of SPARQL end point and APIs on the generated knowledge layer, whereas our pipeline bypasses the limitations of the SPARQL end point to feed directly a set of application persistence tools. In [17] the authors describe a framework based on Apache Spark[15] as data processing tool to manage large scale processing of social network data represented using RDF. Besides the different purpose of the work proposed, the paper focuses on applying specific graph analysis algorithms rather than providing end-users tools for browsing the graph. In [10] big data tools are tested with two benchmark datasets [3]. The benchmark proposes also a set of queries on the generated datasets. However, in our application scenario, we do not consider the RDF base as an application persistence target of queries but we rather use it as a rich and scalable knowledge base that is processed to feed application databases. Furthermore, none of the queries considered in the database considers time, whereas our inference engine is heavily based reasoning over time intervals (see Section 6). Therefore, in this context, while our ETL models data in RDF, we can easily adapt to sink subsets of it into selected alternative databases. In the use case presented in Section 5, at the end of the process we produce JSON objects to be indexed in ElasticSearch and create in SirenSolutions KiBI dashboards (see Figure 2). We chose this tool because it provides unique capability of executing real-time joins over a pool of indexes, enabling seamless relational data exploration and analysis. It is important to stress that we can easily adapt the process to sink data in any type of persistence including RDF triple store, RDMS, OLAP data cubes, or any other legacy system our customer may require. At the best of our knowledge, there is no system that applies a Semantic ETL like the one described in section 2 on real world data for tax assessment purposes.

4. THE *ENTITON* DATA MODEL

The *Entiton* data model is a pragmatic intermediary level of representation between the plain RDF triples (or quads) and the traditional and rigid database tables. Essentially, storing data using the *Entiton* data model is a way of precooking large RDF graphs materializing an entity-centric view over it. The main advantage of adopting the *Entiton* data model is the reduced number of objects involved in data traversing processes (i.e. scans), providing a logical tool to access and manipulate data around objects of interest, rather than statements. Bottom line, it provides a level of granularity that satisfies both requirements of efficiency and expressiveness making life easier for semantic big data application developers. Furthermore, by aggregating *Entiton* objects, it becomes straightforward to build intermediate objects that can be used to provide views over subsets of data. A first version of the *Entiton* implementation is presented in the Data Structure 1, composed by 3 main elements in order of granularity: $EntitonMolecule$,

[11] https://www.elastic.co/
[12] http://siren.solutions/kibi/
[13] https://www.elastic.co/products/kibana
[14] http://optique-project.eu/
[15] http://spark.apache.org/

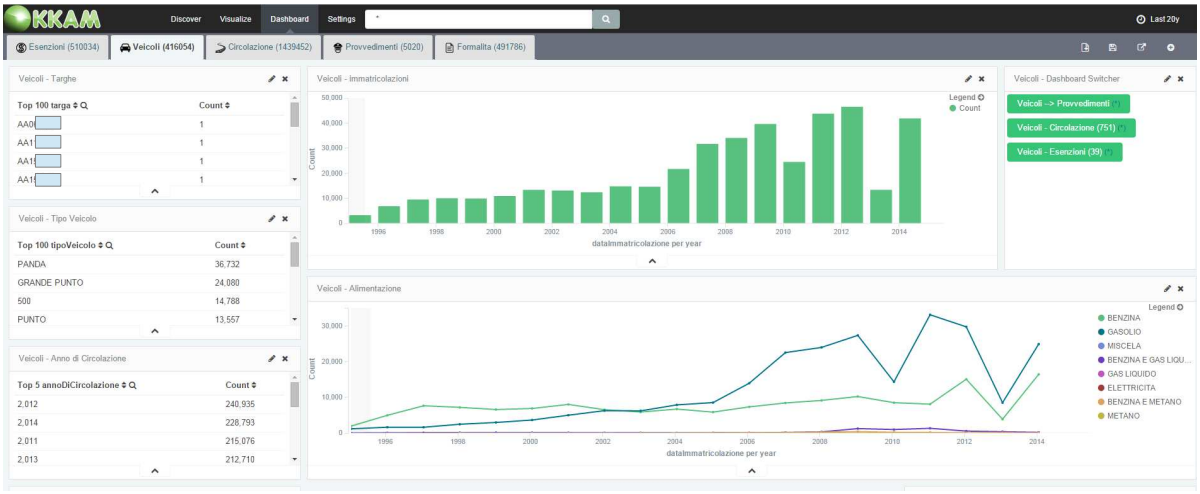

Figure 2: A snapshot of the produced dashboard

EntitonAtom, and *EntitonQuad*. The *EntitonQuad* contains four elements: p (the ontology predicate); o (the object URI); ot (the type of entity identified by the object o); and g (the identifier of the original graph to trace the provenance). The *EntitonAtom* contains four elements: s the URI of the subject as mentioned in the original data source; *oid* the URI provided by the ENS to identify the subject s; *types* a list of types associated with the subject s; and *quads* a list of *EntitonQuad* objects about the subject s. Then finally, the *EntitonMolecule* contains: *atoms* a list of *EntitonAtom*; and r the root *EntitonAtom* for the aggregation of Entiton objects.

```
struct EntitonQuad {
 1: required string p;
 2: required string o;
 3: optional string ot;
 4: optional string g;
}
struct EntitonAtom {
 1: required string s;
 2: optional string oid;
 3: required list<string> types;
 4: required list<EntitonQuad> quads;
}
struct EntitonMolecule {
 1: required EntitonAtom r;
 2: optional list<EntitonAtom> atoms;
}
```

Data Structure 1 Entiton Thrift structure

5. VEHICLE EXCISE DUTY USE CASE

ACI (Automobile Club Italia) is an Italian statutory corporation responsible for promoting and regulating the car sector and to represent car owners' interests in the country. It is therefore in ACI's interest to have a clear view of the status of the car sector in Italy. In particular, they are interested in having an efficient tool for business intelligence and the Vehicle Excise Duty evaders discovery. A first proof-of-concept experiment was executed analyzing data of the Aosta Valley, a small province of about 128.000 inhabitants in the north of Italy. The datasets considered in this experiment consisted of 48 files that sum up to 12 million records about the period 2010-2014 distributed as described in the Table 1.

Dataset	# Records
Italian municipalities	8.092
Fines Register	5.318
Fines	5.319
Exemption	520.874
VED Notification	92.015
VED Payments PRA	1.685.028
VED Payments Region	3.134.642
Vehicles	885.104
Formalities	1.198.940
Litigation	3.705.911
Provisions	447.393
Insurance and Revision	364.134
Pedestrian area Permissions	12.073
Vehicle sales per Municipality	2.465
Company registry	23.376
Total	12.090.675

Table 1: Considered datasets

In about two weeks all these different datasets were loaded in Open Refine, cleaned, mapped towards the defined ontology, and reconciled through the Entity Name System. Each entity represented in the records has been identified either with an ENS IRI or with a local IRI that allows to solve the record in the original data source, and exported as NQuad RDF graph producing about 82 millions of quadruples. At this stage, the RDF NQuad datasets are processed using Apache Flink to build *Entiton* objects and store them in an Apache Parquet file system, clustering information about 7.302.885 entities (among which 184.806 people, 32.577 organization, 8.202 location, 927.860 vehicles, 520.125 exemptions, 3.858.780 VED payments among other events). Noticeably, the largest part of Entitons are events about vehicles (i.e. payments, sales, fines, etc.). Therefore, to estimate the VED evasion rate we need to process all these events and keep them linked to vehicle and owner. In particular, we distinguish between two categories of events: 1) the events defining time intervals during the which citizens can legitimately circulate the owned vehicles (e.g. payments) or they are exempted to pay the VED (e.g. the vehicle is banned from circulation for administrative reasons); and 2) the events falling outside these intervals (or within exemption intervals) and therefore define intervals of time in which there is some irregularity. Hence, based on the information collected we create for each vehicle a sequence of time inter-

vals to asses VED status. It is important to notice that, for each vehicle, it is necessary to consider the whole period of investigation at once. In fact, delayed payments or delays in the notification of exemptions can regularize situations occurred during the previous years. Besides the wide range of exemptions and odd situations one may encounter, the real challenge is to perform such large scale time-based inference in an efficient and effective way. In fact, considering that a small region such as a small province in Italy produced 82M of NQuads and 7.3M Entitons, if we want to support tax investigation on a larger part of the Italian population (only the PRA contains more than 295 million records), we must be ready to scale.

6. SCALABLE TAX ASSESSMENT

The first task of the Tax reasoner is to efficiently scan the *Entiton* knowledge base stored in the Apache Parquet file system to build subsets of object as standard POJOs[16] containing information necessary for the inference. This operation extracts from the *EntitonQuads* from the *EntitonAtom* to assign its value to the POJO that can be efficiently managed by Apache Flink. In particular we extract information about events including unpaid VED notification, vehicle sale contracts, vehicle lifecycle, vehicle owners, VED payments and provisions. Before executing the sequences of checks that would highlight irregular situations, we materialize a set of Entitons about events to complete the information extracted from data. For example, by looking at the first registration date, we generate new exemption intervals for historical cars. Similarly, when we have no information about the car inspections, we generate intervals of exemptions about inspections based on the available first registration date or last known inspection of the vehicle. The inference steps are described in the following paragraphs.

Step 1: Check for Violations of circulation bans. Build exemption intervals dataset sequencing all the events that ban the vehicle from circulating. Run a Full Outer Join relying on the vehicle id between the circulation ban dataset and the datasets that can highlight suspicious of circulation of the vehicle, as for example Vehicle Insurance payments, Fines, Inspections, Pedestrian area parking request, etc. If any join clause is satisfied, we verify whether any of the matching events are contained within the banning interval. If this check is satisfied, the process outputs a set of irregularity statements about the vehicle.

Step 2: Check for VED Payment violation. Build the exemption intervals dataset including: historical vehicle exemptions (computed based on car age), period of permanence in the region, intervals of provisions, and circulation ban as in Step 1. Then build legitimate circulation intervals based on payments events. Compute the union of exemptions and legitimate circulation intervals, and then for each vehicle look for interval gaps comparing it with the taxable period considered (e.g. 5 years between 2010 and 2014). It is important to notice that the taxable period considered for each vehicle is affected by its lifecycle events (e.g. vehicle plat registration, dismissal, and moving out of region) and therefore must be adapted to each vehicle when looking for gaps. If there are sensible gaps (larger than a minimal interval) between the computed taxable period and the union of the legitimate circulation and exemptions, then these account for VED evasions, and irregular events about the vehicle are inferred.

[16]POJO: plain old Java object

Inferred NQuad	11.942.541 (from 54.262.819)
New Entiton	1.605.237 (7.580.106 nq)
Updated Entiton	728.915 (4.362.576 nq)
VED Violations	53572 (13197 new)
Circulation ban Violations	5347 on 506.661 ban
Unregistered contracts	719 on 2423 in municipality

Table 2: Tax reasoner results

Step 3: Cross VED violation with notifications. Given the dataset of VED violations events, we join it with the dataset of Notification events, checking whether the violation has been already notified.

Step 4: Check for Vehicle sales contract registration. Join the dataset of vehicle sales registered at the municipalities offices with the dataset of vehicle sales registered at the national car registry. In fact, the registration of the contract of sales at national car registry (PRA) is subject to an administrative duty. If the contracts of sales are not registered, there is an irregularity in the procedure and registration duties are evaded.

All the inference steps are implemented as an Apache Flink program, although all the sub-parts are meticulously verified using white-box unit tests. Notice that most of the join executes in the process are Full and Left Outer Join[17] to avoid NULL values to affect the data inference and merging process. In fact, relying on simple Join operations, if someone did not every paid VED for a vehicle, we would not be able to discover it. This applies the OPTIONAL principles of SPARQL[18]. The results of the execution of the aforementioned process on the cluster described in Section 7 are presented in Table 2. We discovered 13197 new VED evasion cases, and discovered 5347 circulation ban violations. Noticeably, circulation ban violations could not be estimated with the legacy system. The execution time of the overall inference process is in average 5 minutes and 45 seconds.

The choice of relying on a highly parallel data processing environment, rather than a triple store and SPARQL, enables scalability and increase robustness of the process. In fact, working with a Java process and POJOs allows to intercept and correct on the run part of the long tail of errors typical of large-scale data processing. However, the inference process still outputs NQuads statements creating a novel data source that can be used to extend the Entiton dataset. Therefore, ontology and RDF are still extremely useful tools to persist and maintain a large scale knowledge base on a distributed file system.

For the sake of this paper, in order to make the experiments reproducible, we created a synthetic dataset resembling the original one that cannot be disclosed for obvious privacy reasons. As labels do not make sense in this case, and there is not need of supporting real tax investigators, we removed all labels. We uploaded a compressed folder containing an Apache Flink distribution, the jar dependencies of the inference program, the synthetic dataset, and some instructions to run the experiment at http://dev.okkam.it/SBD2016/okkam-sbd2016.zip.

7. CLUSTER DESCRIPTION

[17]https://ci.apache.org/projects/flink/flink-docs-release-0.10/apis/dataset_transformations.html#outerjoin

[18]https://www.w3.org/2008/07/MappingRules/StemMapping

All the software was tested on a cluster of 6 consumer machines with the following characteristics: CPU QuadCore Intel I7 4770R 3.20 GHz, 16GB of RAM, 256GB SSD disk + 1T 7200rpm HD, running a Linux Ubuntu 12.04.03LTS with Cloudera 5.1.3, Hadoop 2.3.0- cdh5.1.3, Apache Flink 0.10.1 on Oracle Java 1.8.0.72. The same 6 machines are used for the Entity Name System, that uses Apache Solr 4.10.1 and Apache Tomcat 7.0.55 on 2 nodes, and Apache HBase 0.98 RegionServer and HDFS DataNode on 3 nodes, and 1 node with master elements (i.e. NameNode, Zookeeper, and other Cloudera Management services).

8. CONCLUSIONS

In this paper we presented a data management system that combines: 1) semantic technologies, 2) custom Open Refine as an effective data cleaning tool, 3) an ENS as a scalable reconciliation service, 4) Apache Flink as a big data processing framework, and 4) Siren Solution KiBI for seamless data intelligence. We used the system to support a tax assessment use case, testing it with the data of Aosta Valley in Italy. The experiment was successful, and the technology stack is now installed within the ACI Informatica data centers, supporting tax investigation for millions of citizen in other regions of Italy. We designed a scalable tax inference engine that can be deployed in a distributed setting, and is capable of inferring 11 million new statements on a dataset of 55 million triples taking in average 5 minutes and 45 seconds on a small cluster of 6 consumer machines. Next evolutions will develop two directions: 1) improving the usability of Open Refine to reduce the human effort of data cleaning process; 2) automatize those steps that do not strictly require human supervision. Disrupting the traditional semantic data management process, we leverage semantic technologies to model and map heterogeneous data sources and to efficiently reconcile entities, but we rely on scalable Java programs to process data and effectively deal with the long tail of errors and oddities. We believe that thanks to this combination of Semantic and Big Data technologies we are now walking the last mile for the creation of real world semantic applications.

9. REFERENCES

[1] G. Adinolfi, P. Bouquet, M. Zeni, and S. Bortoli. Sicras: Semantic (big) knowledge for fighting tax evasion and supporting social policy making. In A. Polleres, A. Garcia, and R. Benjamins, editors, *Proceedings of ISWC2014 Industry Track Abstracts*, October 2014.

[2] A. Berson and S. J. Smith. *Data Warehousing, Data Mining, and Olap*. McGraw-Hill, Inc., New York, NY, USA, 1st edition, 1997.

[3] C. Bizer and A. Schultz. The berlin sparql benchmark. *International Journal on Semantic Web and Information Systems*, 5(1):1–24, 2009.

[4] S. Bortoli. *Knowledge Based Open Entity Matching*. PhD thesis, International Doctoral School in ICT of the University of Trento (Italy), 2013.

[5] S. Bortoli, P. Bouquet, and B. Bazzanella. An identification ontology for entity matching. In *On the Move to Meaningful Internet Systems: OTM 2014 Workshops*, volume 8842 of *Lecture Notes in Computer Science*, pages 587–596. Springer Berlin Heidelberg, 2014.

[6] S. Bortoli, P. Bouquet, and B. Bazzanella. Okkam synapsis: a community-driven hub for sharing and reusing mappings across vocabularies. In *SWCS'14 Proceedings of the Third International Conference on Semantic Web Collaborative Spaces*, volume Volume 1275, pages 27–37, 2014.

[7] P. Bouquet, T. Palpanas, H. Stoermer, and M. Vignolo. A conceptual model for a web-scale entity name system. In *Asian Semantic Web Conference (ASWC)*, Shanghai, China, 2009.

[8] P. Bouquet, H. Stoermer, C. Niederee, and A. Mana. Entity name system: The backbone of an open and scalable web of data. In *Proceedings of the IEEE International Conference on Semantic Computing, ICSC 2008*, pages 554–561. IEEE Computer Society, August 2008.

[9] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, A. Poggi, M. Rodriguez-Muro, R. Rosati, M. Ruzzi, and D. F. Savo. The mastro system for ontology-based data access. *Semant. web*, 2(1):43–53, Jan. 2011.

[10] P. Cudré-Mauroux, I. Enchev, S. Fundatureanu, P. Groth, A. Haque, A. Harth, F. L. Keppmann, D. Miranker, J. F. Sequeda, and M. Wylot. Nosql databases for rdf: An empirical evaluation. In *The Semantic Web - ISWC 2013*, volume 8219 of *LNCS*, pages 310–325. Springer Berlin Heidelberg, 10 2013.

[11] M. A. Hernández and S. J. Stolfo. Real-world data is dirty: Data cleansing and the merge/purge problem. *Data Min. Knowl. Discov.*, 2(1):9–37, Jan. 1998.

[12] E. Kharlamov, N. Solomakhina, O. L. Özçep, D. Zheleznyakov, T. Hubauer, S. Lamparter, M. Roshchin, A. Soylu, and S. Watson. How semantic technologies can enhance data access at siemens energy. In *Proceedings of the 13th International Semantic Web Conference - Part I*, ISWC '14, pages 601–619, New York, NY, USA, 2014. Springer-Verlag New York, Inc.

[13] B. Neumayr, S. Anderlik, and M. Schrefl. Towards ontology-based olap: Datalog-based reasoning over multidimensional ontologies. In *Proceedings of the Fifteenth International Workshop on Data Warehousing and OLAP*, DOLAP '12, pages 41–48, New York, NY, USA, 2012. ACM.

[14] E. Raad and J. Evermann. Is ontology alignment like analogy? – knowledge integration with lisa. In *Proceedings of Symposium On Applied Computing (SAC), Korea, Republic Of (2014)*, 2014.

[15] T. Shah, F. Rabhi, and P. Ray. Investigating an ontology-based approach for big data analysis of inter-dependent medical and oral health conditions. *Cluster Computing*, 18(1):351–367, Mar. 2015.

[16] R. Verborgh and M. D. Wilde. *Using OpenRefine*. PACKT Publishing, 2013.

[17] Z. Xu, W. Chen, L. Gai, and T. Wang. Sparkrdf: In-memory distributed rdf management framework for large-scale social data. In *Web-Age Information Management, Proceedings of 16th International Conference*, volume 9098 of *LNCS*, pages 337–349. Springer International Publishing, June 2015.