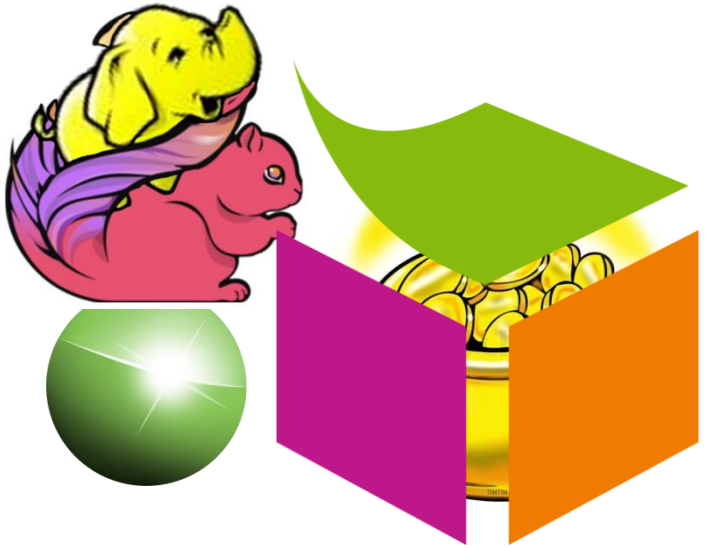
Semantic Big Data 2016

1st of July, w/ ACM SIGMOD 2016 in San Francisco, USA

SICRaS
ALYSSO • ENGINEERING • OKKAM

Semantic Big Data for Tax Assessment



Stefano Bortoli **@stefanobortoli**
bortoli@okkam.it (bortoli@disi.unitn.it)

Paolo Bouquet **@paolobouquet**
bouquet@okkam.it (bouquet@disi.unitn.it)

Flavio Pompermaier **@fpompermaier**
pompermaier@okkam.it

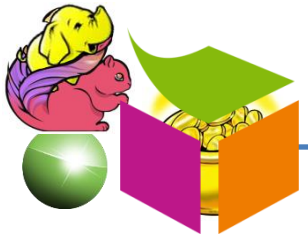
Andrea Molinari **@molinariandrea**
andrea.molinari@unitn.it

OKKAM
EMPOWERING THINGS

The company (briefly)

- Okkam is
  - a SME based in Trento, Italy.
  - Started as joint spin-off of the University of Trento and FBK (2010)
- Okkam core business is
  - large-scale data integration using semantic technologies and an Entity Name System
- Okkam operative sectors
  - Services for public administration
  - Services for restaurants (and more)
  - Research projects
    - EU FP7, EU H2020, and Local agencies

Our toolbox



MySQL
PostgreSQL
Virtuoso Universal Server
Apache Solr
kibana
Parquet
APACHE HBASE
elastic
hadoop HDFS
mongoDB
lucene
SIREn SOLUTIONS
AVRO
JSON-LD
OPEN Refine
KRYO
Jena
protobuf
Protocol Buffers - Google's data interchange format
Apache Thrift ™
protégé
W3C OWL2
Java ORACLE
Maven
eclipse
spring by Pivotal
Jenkins
sonarqube
artifactory
WSO2
akka

Hardware-wise

8 x Gigabyte Brix
16GB RAM
256GB SSD
1T HDD
Intel I7 4770 3,2Ghz
+
1 Gbit Switch

- We compete with expensive data warehouse solutions
  - e.g. Oracle Exadata Database Machines, IBM Netezza, etc.
- Test on small machines fosters optimization
  - If you don't want to wait, make your code faster!
- Our code is ready to scale, without big investments
- Fancy stuff can be done without large investments in HW

Using semantics at scale

Entiton data model

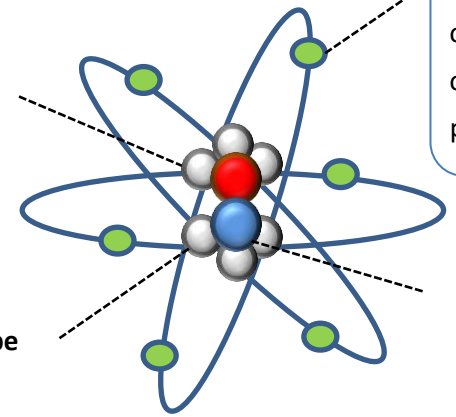
First Name	Last Name	Address	City	Age
Mickey	Mouse	123 Fantasy Way	Anaheim	73
Bat	Man	321 Cavern Ave	Gotham	54
Wonder	Woman	987 Truth Way	Paradise	39
Donald	Duck	555 Quack Street	Mallard	65
Bugs	Bunny	567 Carrot Street	Rascal	58
Wiley	Coyote	999 Acme Way	Canyon	61
Cat	Woman	234 Purrfect Street	Hairball	32
Tweety	Bird	543	Itotltaw	28

Records

Database record

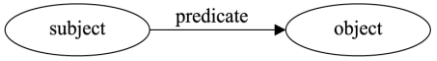
Subject
Global IRI

RDF Type



Quad
predicate
object
object Type
provenance IRI

Subject
local IRI



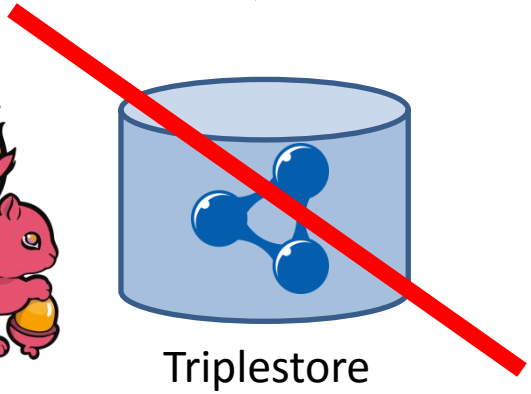
Source: W3C (2004)

RDF statement

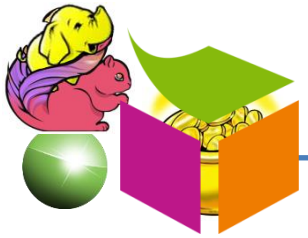
Expensive datawarehouse

NOSQL
+ Indexes

+



Triplestore



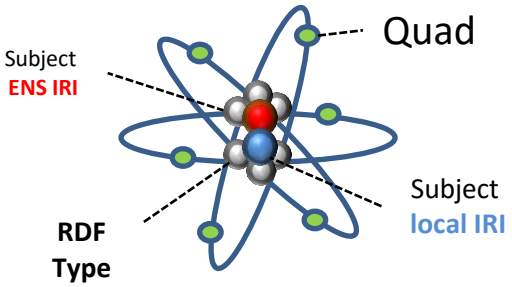
Entiton using Parquet+Thrift

```
namespace java
it.okkam.flink.entitons.serialization.thrift

struct EntitonQuad {
        1: required string p;   //pred
        2: required string o;   //obj
        3: optional string ot; //obj-type
        4: required string g;   //sourceIRI
}

struct EntitonAtom {
        1: required string s;                    //local-IRI
        2: optional string oid;                  // ens-IRI
        3: required list<string> types;          //rdf-types
        4: required list<EntitonQuad> quads;     // quads
}

struct EntitonMolecule {
        1: required EntitonAtom r;               //root atom
        2: optional list<EntitonAtom> atoms;     //other atoms
}
```



Tax Assessment use case

Automobile Club d'Italia

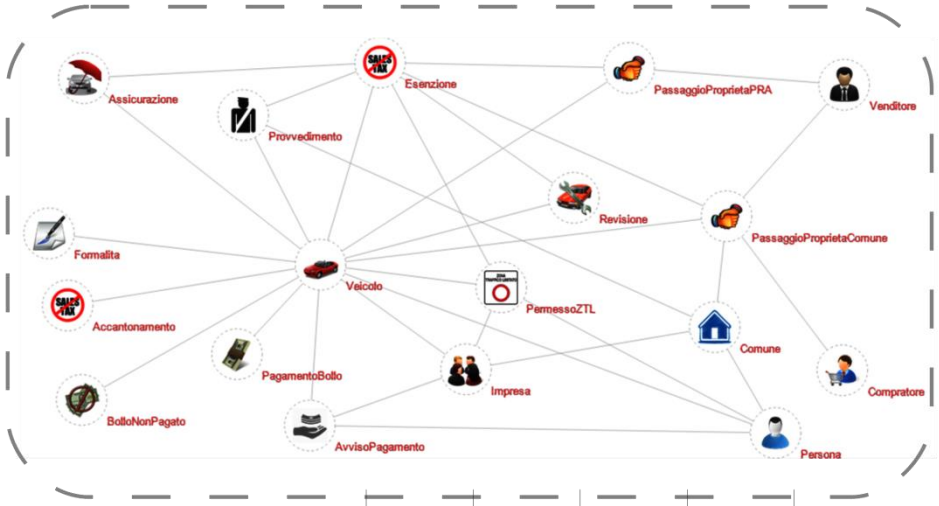
Pilot project for ACI and Val d'Aosta

- Objectives are to investigate:
  1. Who did not pay Vehicle Excise Duty?
  2. Who did not pay Vehicle Insurance?
  3. Who skipped Vehicle Inspection?
  4. Who did not pay Vehicle Sales Taxes?
  5. Who violated circulation ban?
  6. Who violated exceptions to the above?

Dataset: **15 data sources** for 5 year with **12M records** about **950k vehicles** and **500k subjects** for a total of **82M NQuad** statements

Challenge: **consider events (time)** and **infer implicit information**.

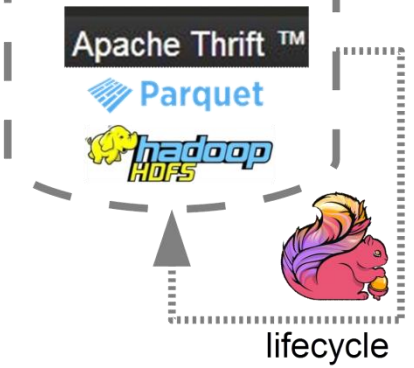
Semantic Big Data ETL

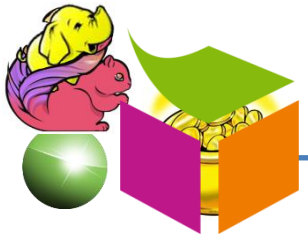


Fines
Car registry
Payments
Sales registry
Exceptions
Warnings
Formalities
Inspections
Insurances
Court cases

Semantic ETL OPEN Refine

Acquisition
Normalization
Mapping
Classification
Riconciliation





Tax Assessment steps

- Load Entitons into POJOs
- Materialized implicit info, e.g.:
  - Car inspection and other lifecycle dates
  - Classify historical vehicles (as they are exempted)
- Check for circulation ban violations
  - Build the circulation ban for all vehicles
  - Join intervals with all events unusual for ban period and materialize irregularity
- Check VED payment violation
  - Compute the union of legitimate circulation and all exemptions
  - Check for gaps considering the assessment period and materialize irregular intervals above a threshold as VED violations
- Cross VED violation with notifications

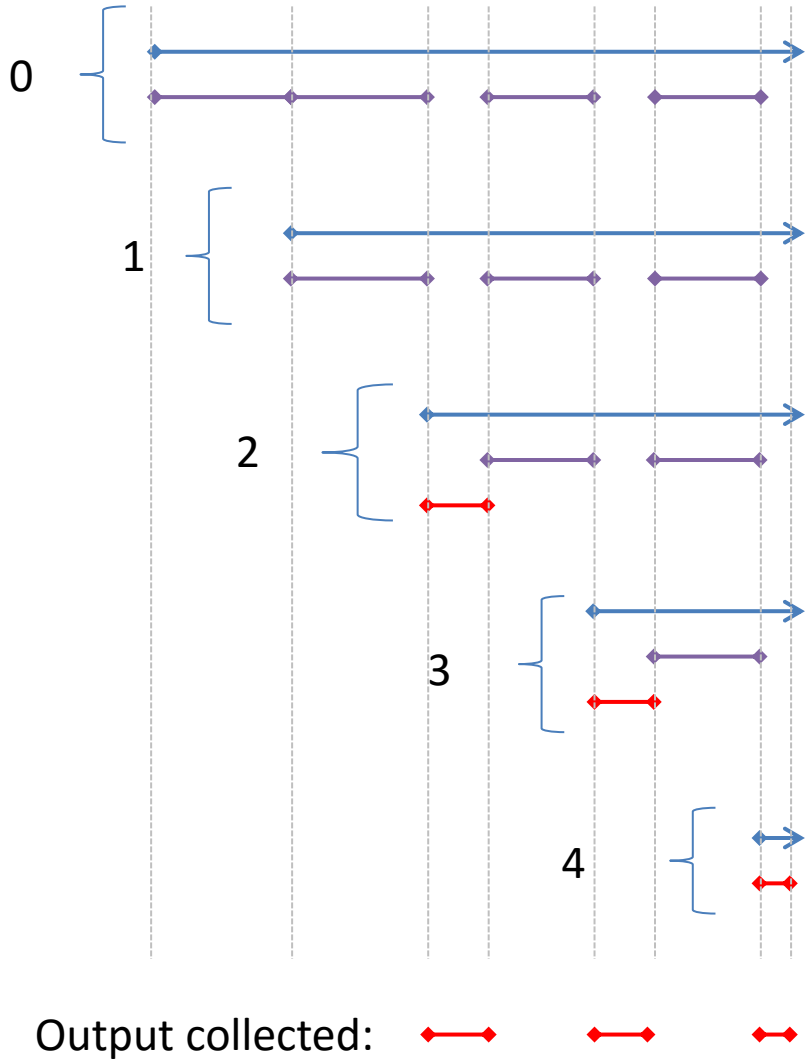
Gap detection for one vehicle

All legitimate events are represented as a sorted list of merged Joda Time Intervals to be verified against the assessment period

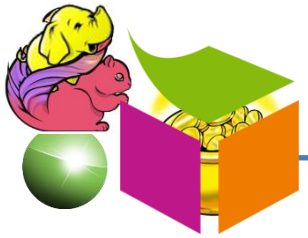
The algorithm iteratively checks each interval start and end to be contained in the assessment period, moving ahead the start of the assessment period when everything is correct

If there is difference between the start of the assessment period and the start of the next legitimate interval, then a gap interval is created

If legitimate interval ends before end of the assessment period, then a gap interval is created

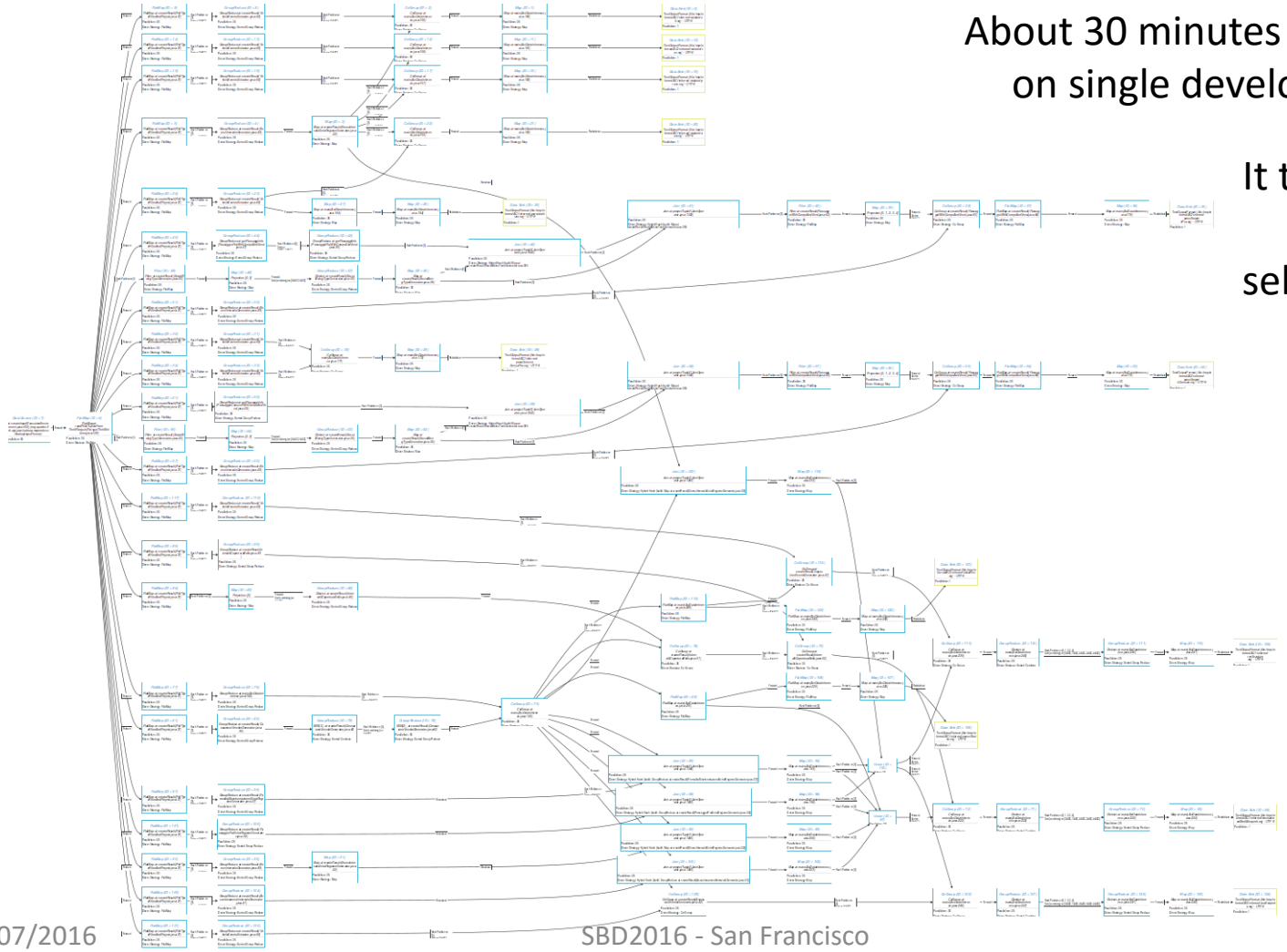


Tax Reasoner



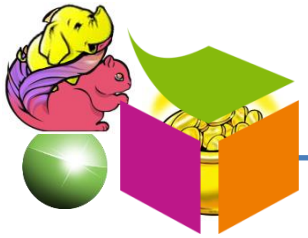
Automobile Club d'Italia

Temporal Inference Execution Plan



About 30 minutes ETA with SSD on single developer machine

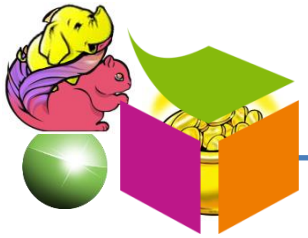
It took **1 DAY** to perform the select query for one of the sources!!



Inference results

- On “cluster” the average execution time was ~6 min
  - 11.9M new NQuad statements inferred
  - 1.6M new entiton objects
  - 725k entitons updated
  - 53k VED violations
  - 5k circulation ban violations
- Between 11.3% and 15.5% of vehicle had issues with VED
- Near 7,6% of vehicles with car inspection issues
- Near 9.3% of vehicles circulated without insurance

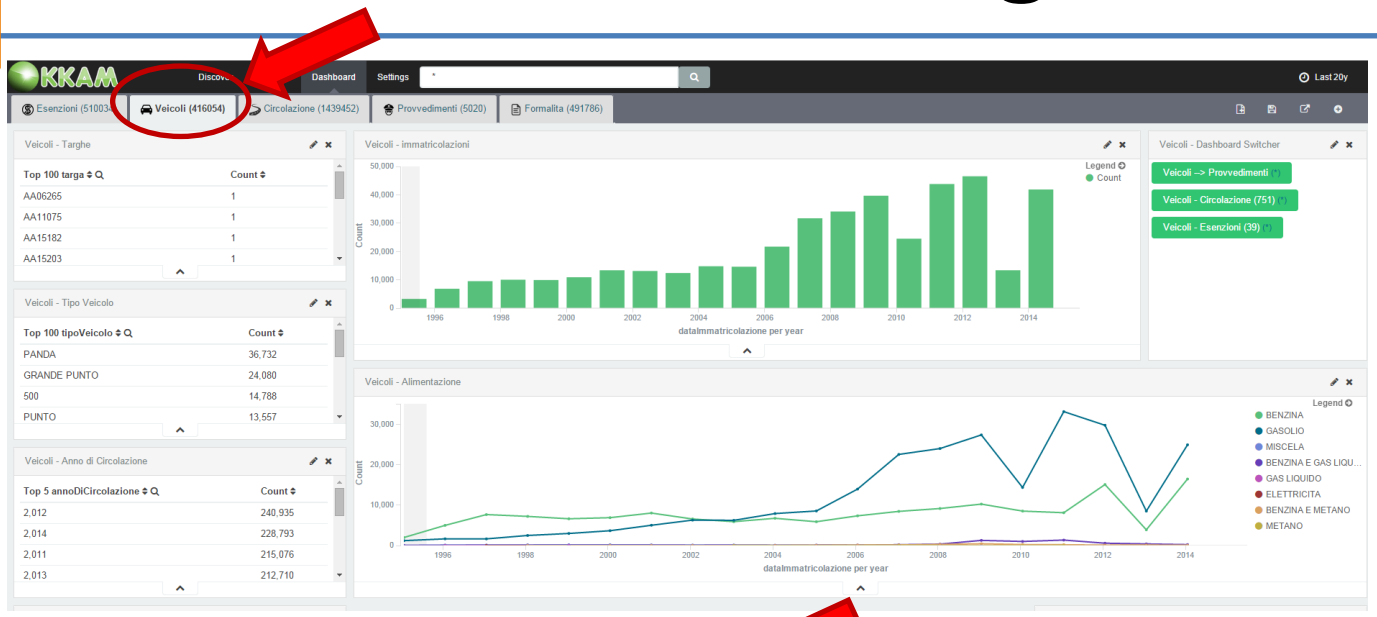
Clerical review of some cases verified soundness of the inference process, improving of about 1% with respect to in place systems running on slow and expensive data warehouse solutions.

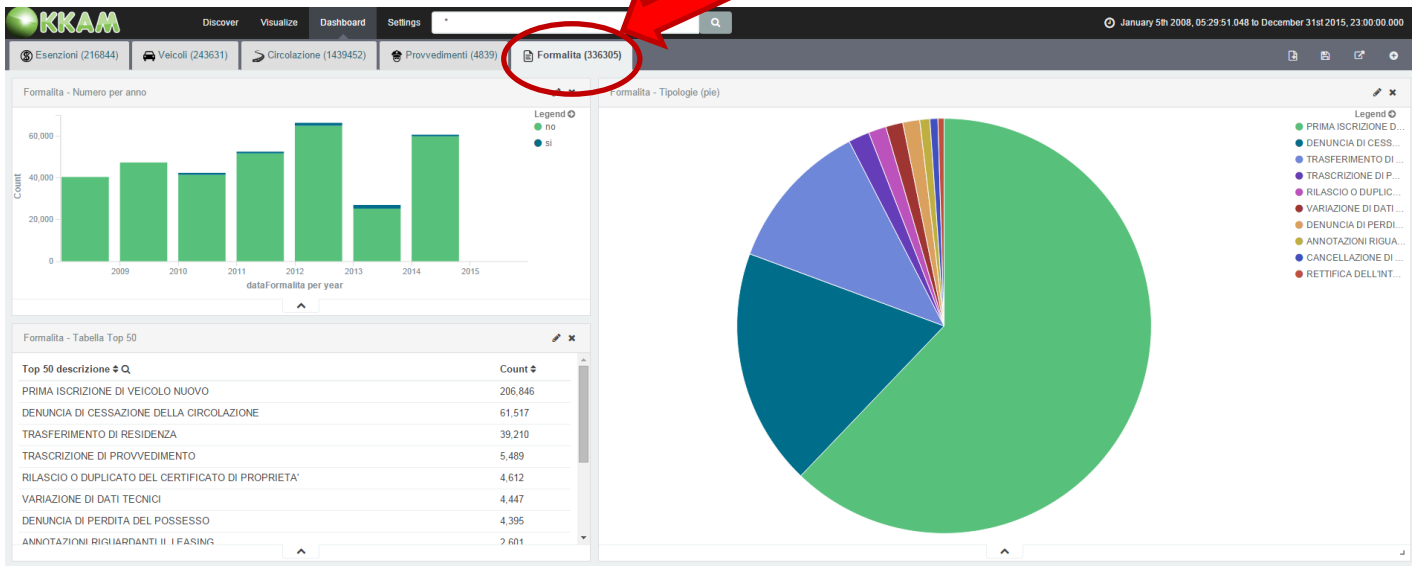


From Entitons to RDF Intelligence

- Each Entiton object is processed to produce a JSON document, exploring relational paths when required
  - e.g. to associate a plate number to VED evasion event entiton, we need to get the vehicle entiton, and therefore its plate
- Entiton JSON objects are grouped in files according to entity type defined in the ontology
- JSON files are loaded in **ElasticSearch** with **LogStash**, creating one index per entity type in the ontology
- We configure the relations among the indexes in **SirenSolution KiBi** to allow multi-dimensional and cross-dashboard data exploration
- We create the dashboards presenting the data

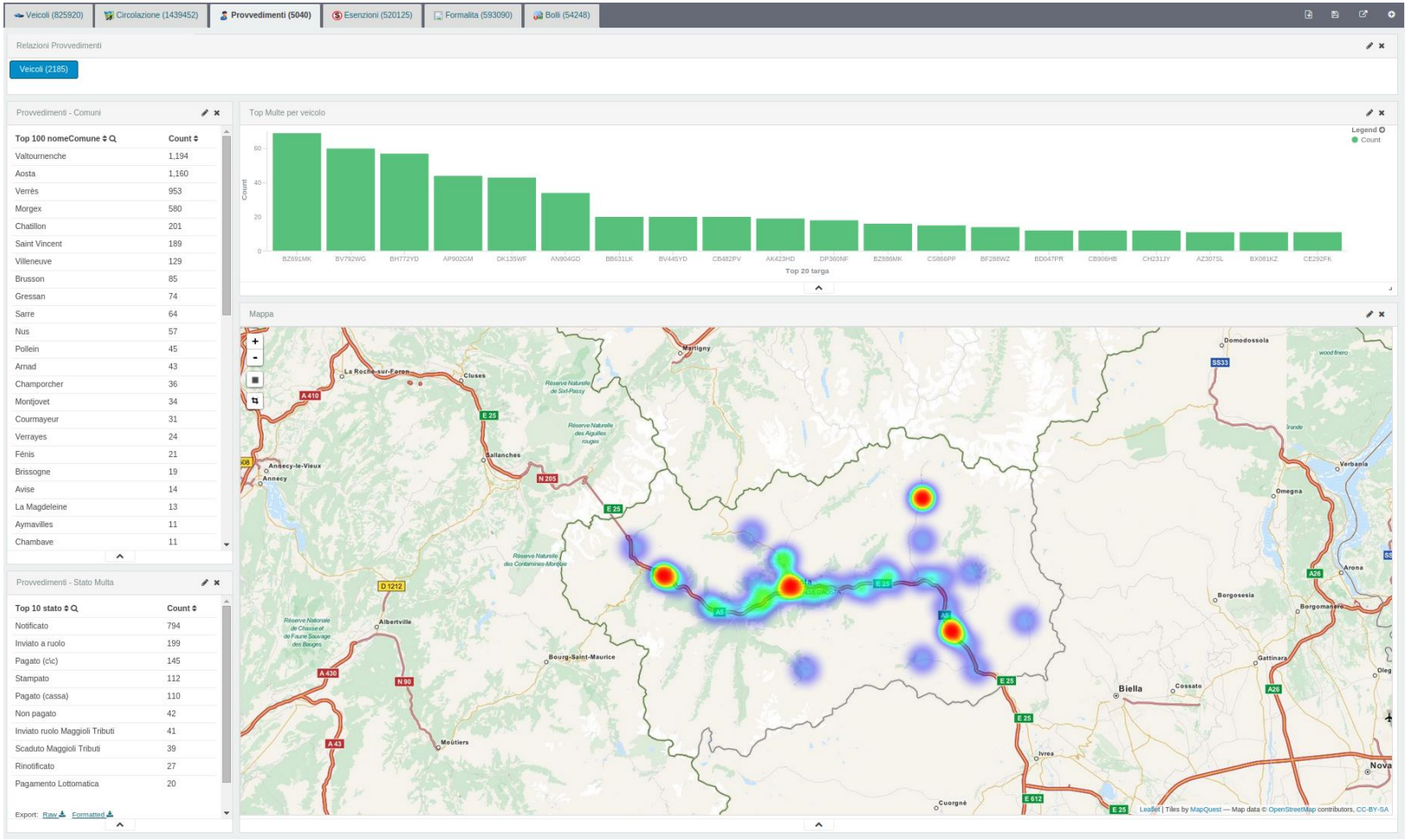
RDF Data Intelligence





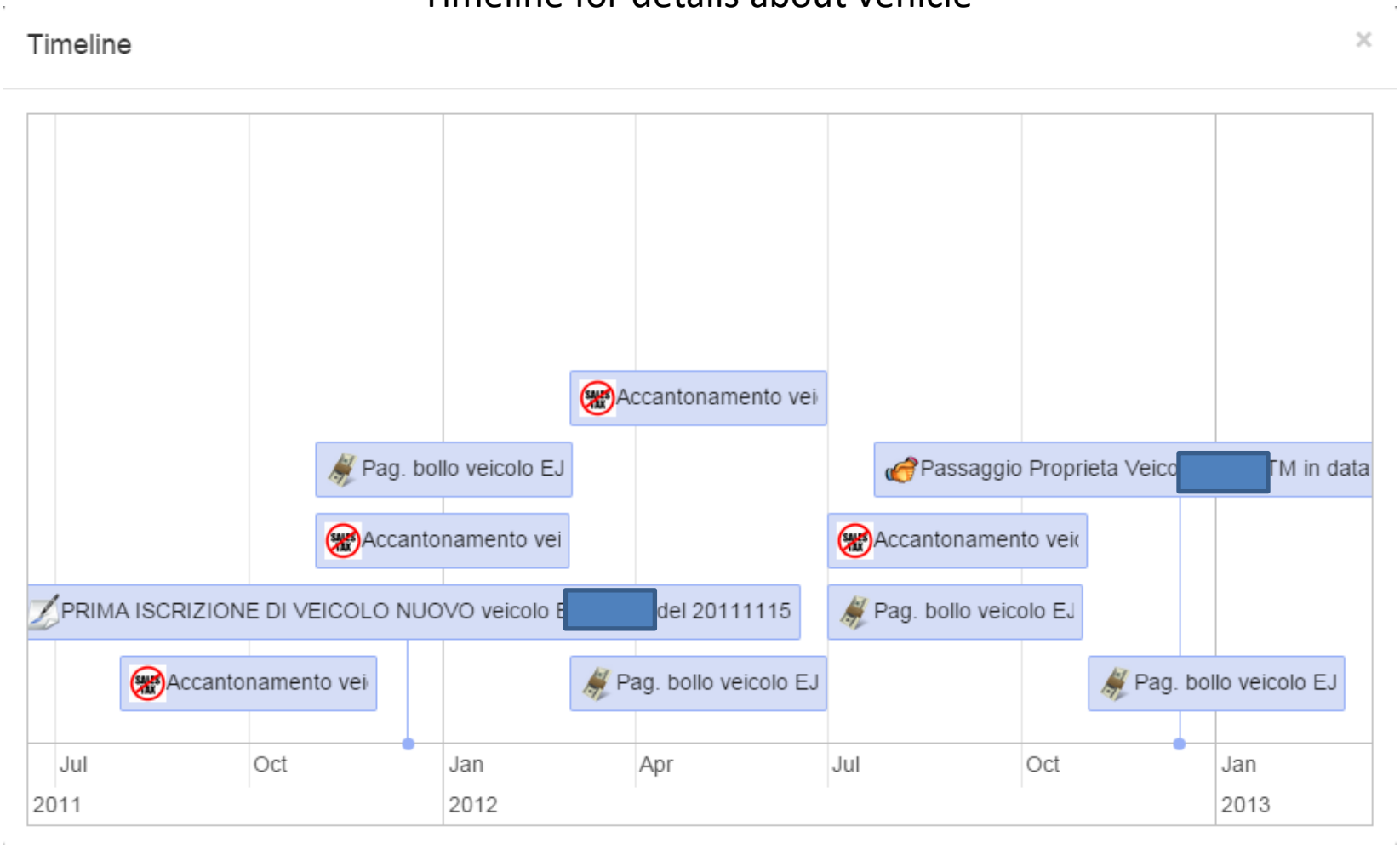
RDF Data Intelligence

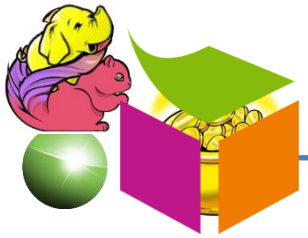
Geospatial indicators



RDF Data Intelligence

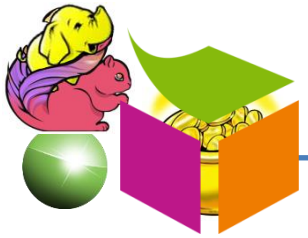
Timeline for details about vehicle





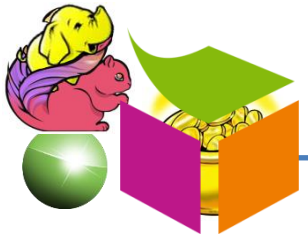
Technical Lessons learned

- **Reversing String Tuples ids** leads to performance improvements of joins
- When you make joins, ensure **distinct dataset keys**
- **Reuse objects** to reduce the impact of garbage collection
- When writing Flink jobs, start with small and debuggable **unit tests first**, then run it on the cluster on the entire dataset (waiting for big data debugging methods result of Marcus Leich work at Technical University of Berlin - DE)
- **Serialization matters**: less memory required, less gc, faster data loading → faster execution
- HD speed matters when RAM is not enough, **SSD rulez**
- **Apache Parquet rulez**: self-describing data, push-down filters



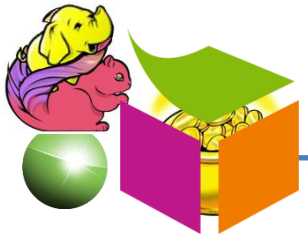
Future works

- Benchmark Entiton serialization models on Parquet (Avro vs Thrift vs Protobuf)
- Manage declarative data fusion policies
  - *a-la* LDIF: http://ldif.wbsg.de/
- Define an algebra for entiton operations (e.g. merge, project, select, filter, reconcile, smush)
- Manage provenance metadata for inferred data
- Try out Cloudera Kudu
  - novel Hadoop storage engine addressing both bulk loading stability, scan performance and random access
  - https://github.com/cloudera/kudu



Conclusions

- We think we are walking along the “last mile” towards real world **enterprise Semantic Applications**
- Combining big data and semantics allows us to be flexible, expressive and, thanks to Flink, very scalable at very competitive costs
- Apache Flink gives us the leverage to shuffle data around without much headache
- We proved cool stuff can be done in a simple and efficient way, with the right tools and mindset
- We need to automatize the process, but in this domain it does not sound too problematic



Thanks for your attention

Any Questions?