Tutorial on RDF Stream Processing

M. Balduini, J-P Calbimonte, O. Corcho,
D. Dell'Aglio, E. Della Valle

http://streamreasoning.org/rsp2014

SPARQLStream: Ontology-based access to data streams

Jean-Paul Calbimonte, Oscar Corcho
jp.calbimonte@upm.es, ocorcho@fi.upm.es
http://www.oeg-upm.net/

Share, Remix, Reuse — Legally

- This work is licensed under the Creative Commons Attribution 3.0 Unported License.
- **You are free:**

to Share — to copy, distribute and transmit the work

to Remix — to adapt the work

- **Under the following conditions**

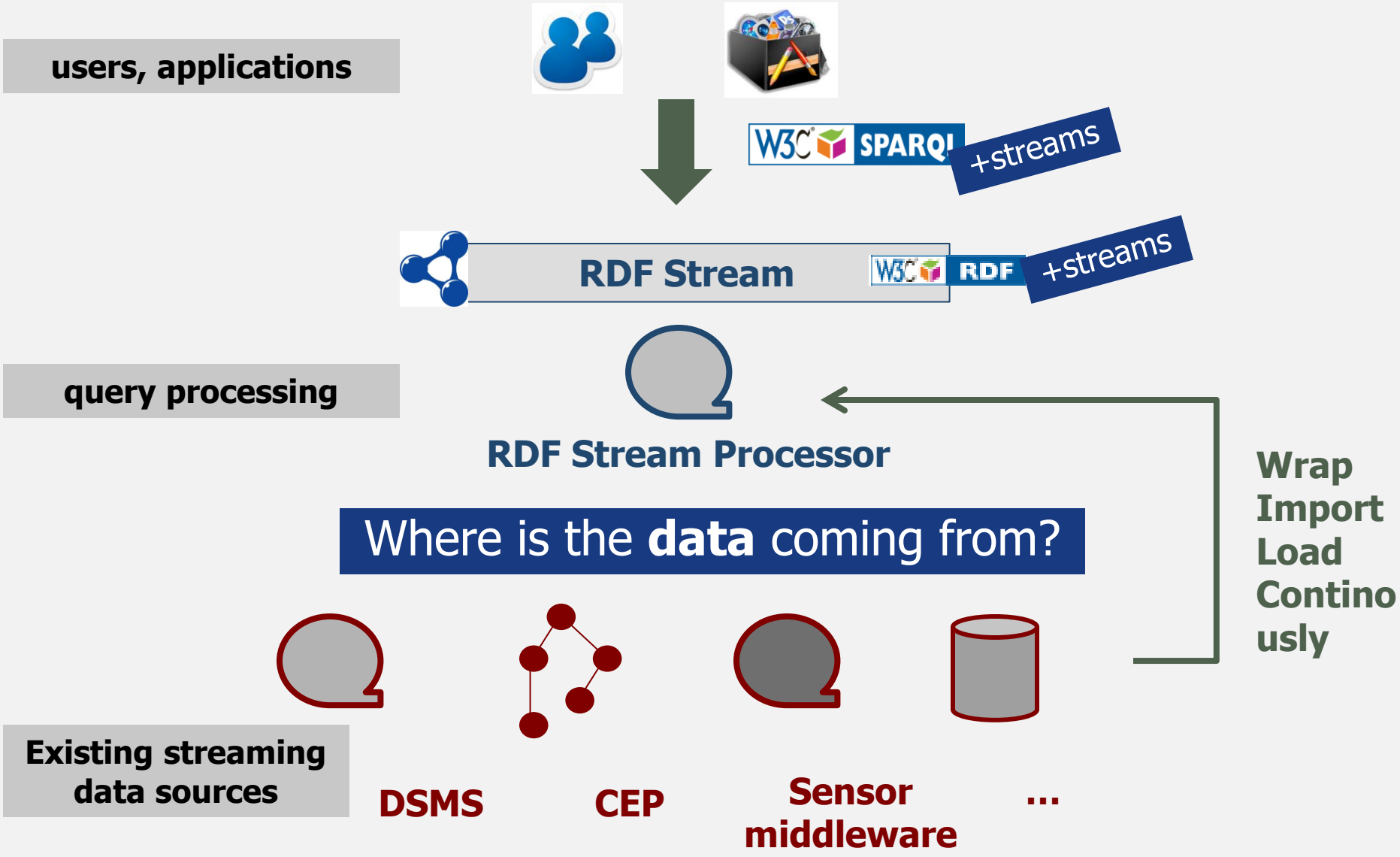
Attribution — You must attribute the work by inserting
  - "[source http://streamreasoning.org/sr4ld2013]" at the end of each reused slide
  - a credits slide stating
    - These slides are partially based on "Streaming Reasoning for Linked Data 2013" by M. Balduini, J-P Calbimonte, O. Corcho, D. Dell'Aglio, E. Della Valle, and J.Z. Pan http://streamreasoning.org/sr4ld2013

- To view a copy of this license, visit http://creativecommons.org/licenses/by/3.0/

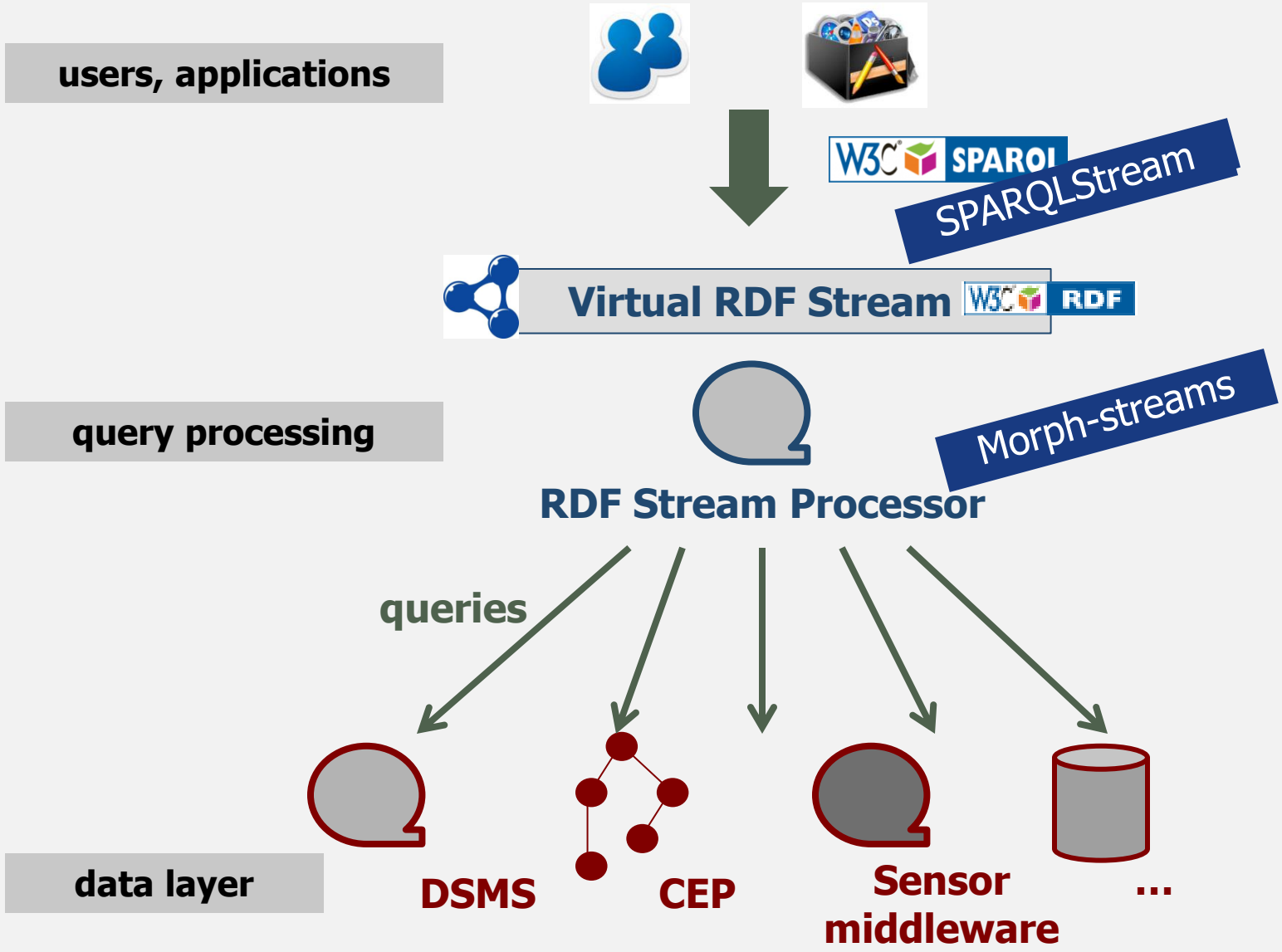
SPARQLStream

- Virtual RDF views over data streams
- Ontology-based access to data streams
  - Examples
  - Architecture
  - Underlying query processors
- SPARQLStream language
- Query rewriting
  - R2RML mappings
- Resources

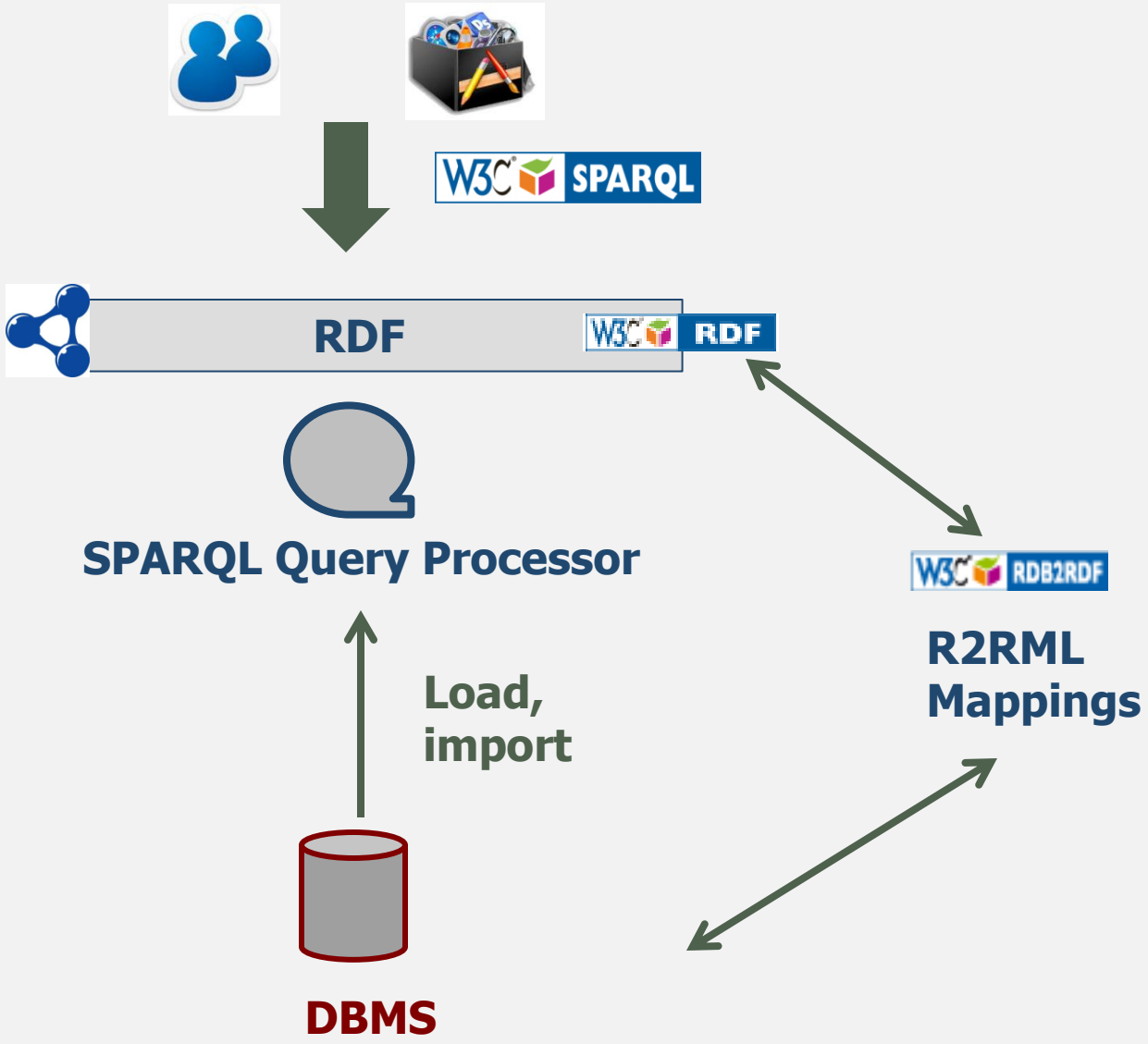
Querying RDF Streams



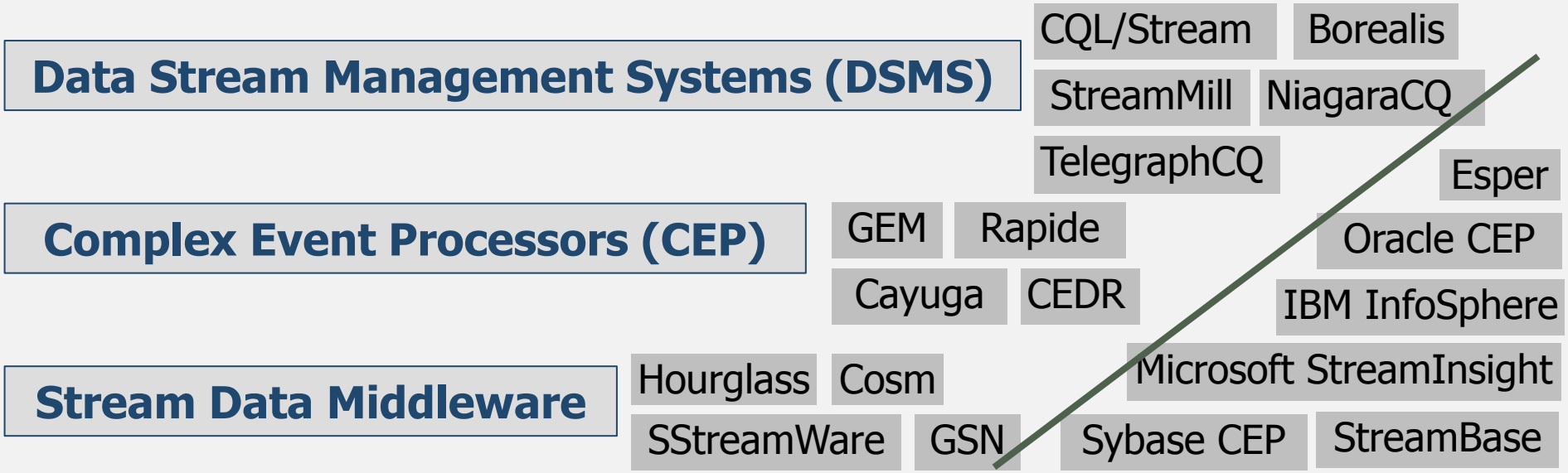
Virtual RDF views over data streams



Already seen somewhere…?



Stream Processor Implementations



Diverse **query languages**

Different **query capabilities**

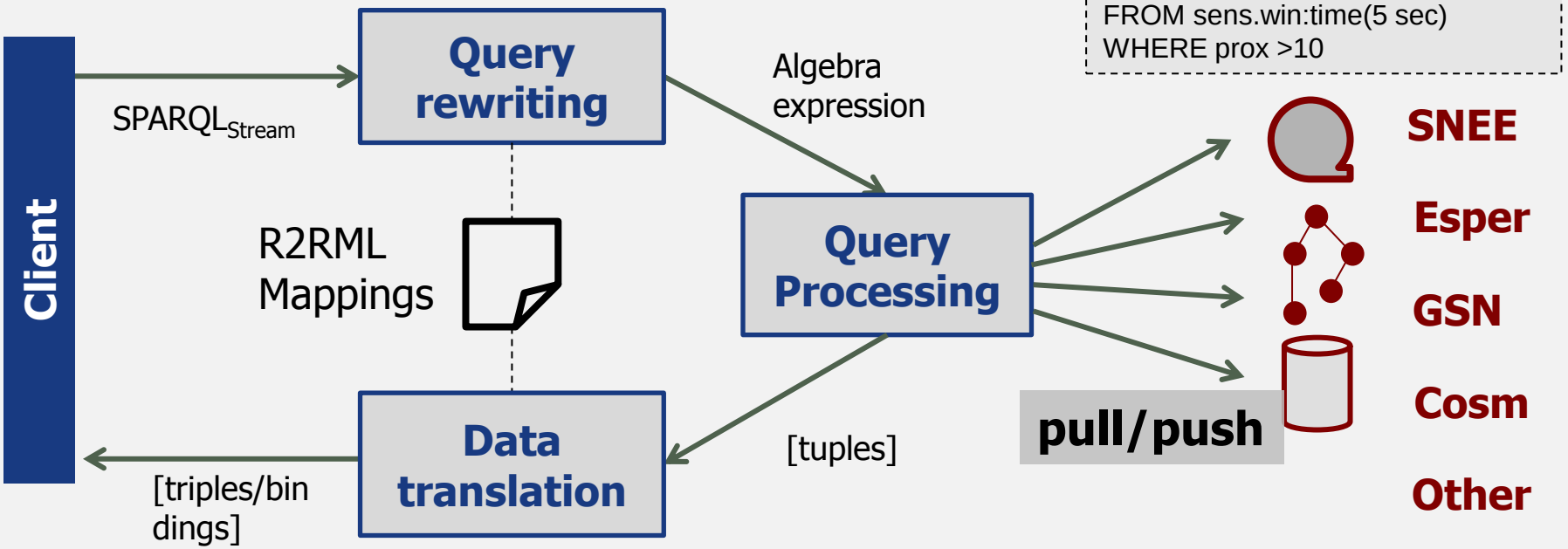
Different **query models**

Morph-streams: Overview

```
SELECT ?proximity
FROM STREAM
<http://streamreasoning.org/SensorReadings.srdf> [NOW–5
S]
WHERE {
  ?obs a ssn:ObservationValue;
       qudt:numericalValue ?proximity;
  FILTER (?proximity>10) }
```

$\pi_{timed,prox}$
$\sigma_{prox>10}$
$\omega_{5\ Seconds}$
sens

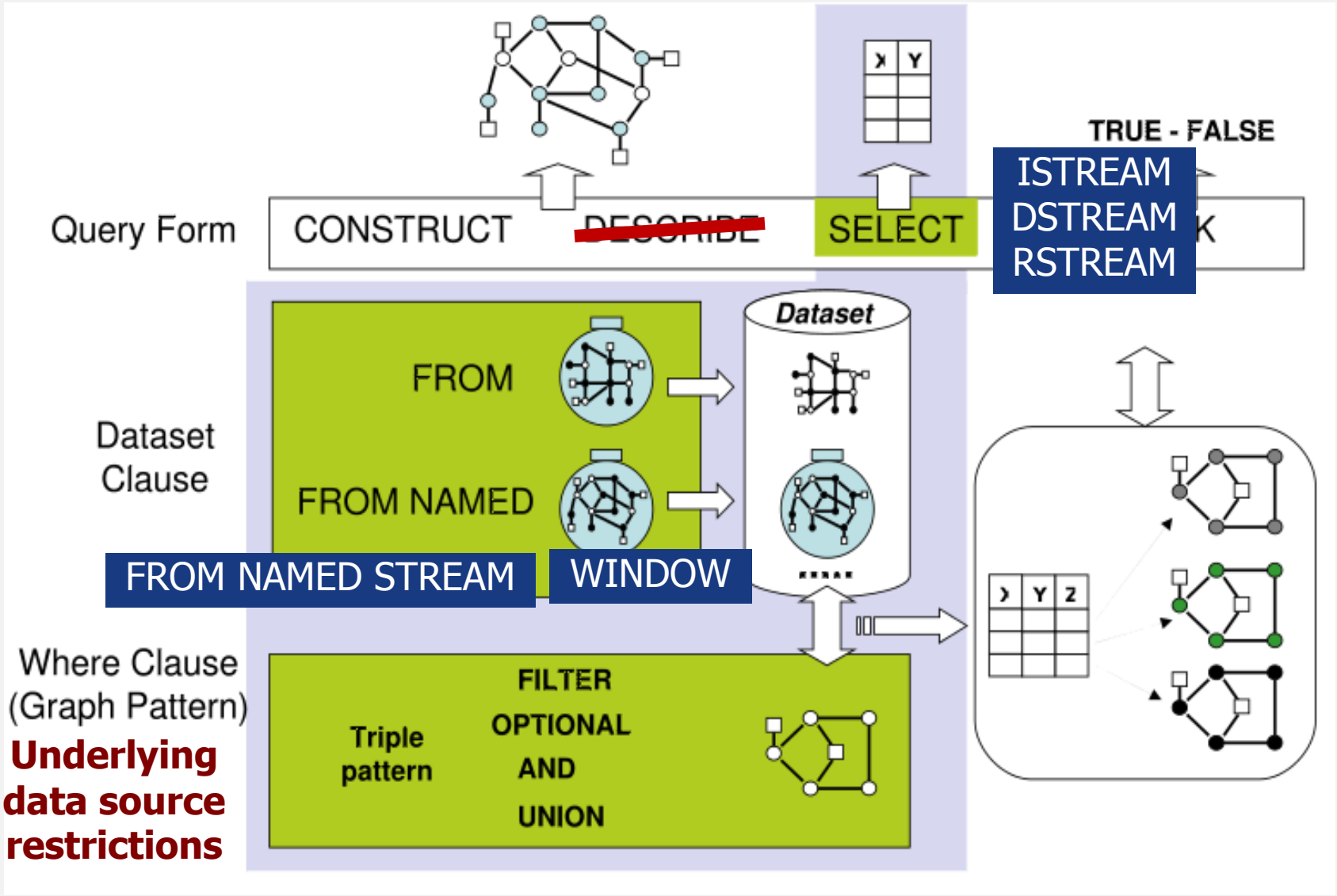
```
SELECT prox
FROM sens.win:time(5 sec)
WHERE prox >10
```



Morph-streams procesing SPARQL$_{Stream}$ queries

https://github.com/jpcik/morph-streams

SPARQLStream Language



SPARQLStream: examples

SPARQL$_{Stream}$

```
PREFIX sr4ld: <http://www.streamreasoning.org/ontologies/socialsensor,owl#>
SELECT ?room
FROM NAMED STREAM <http://www.streamreasoning.org/streams/socialsensor.srdf> [NOW-10 S]
WHERE {
    ?obs sr4ld:observedBy ?sensor.
    ?obs sr4ld:where  ?room.
}
```

All rooms where something was observed in the last 10s

```
PREFIX sr4ld: <http://www.streamreasoning.org/ontologies/socialsensor,owl#>
SELECT (COUNT(?person) AS ?nmb) ?room
FROM NAMED STREAM <http://www.streamreasoning.org/streams/socialsensor.srdf> [NOW-10 S]
WHERE {
    ?obs sr4ld:who ?pers.
    ?obs sr4ld:where  ?room.
}
GROUP BY ?room
```

Number of persons observed in each room in the last 10s

SPARQLStream Language

- *NamedStream* → 'FROM' ['NAMED'] 'STREAM' *StreamIRI* '[' *Window* ']'
- *Window* → 'NOW-' *Integer TimeUnit* [*UpperBound*] [*Slide*]
- *UpperBound* → 'TO NOW-' *Integer TimeUnit*
- *Slide* → 'SLIDE' *Integer TimeUnit*
- *TimeUnit* → 'MS' | 'S' | 'MINUTES'| 'HOURS' | 'DAY'

```
SELECT ISTREAM ?room
FROM NAMED STREAM <http://www.streamreasoning.org/streams/socialsensor.srdf> [NOW-10 S]
WHERE {…
```

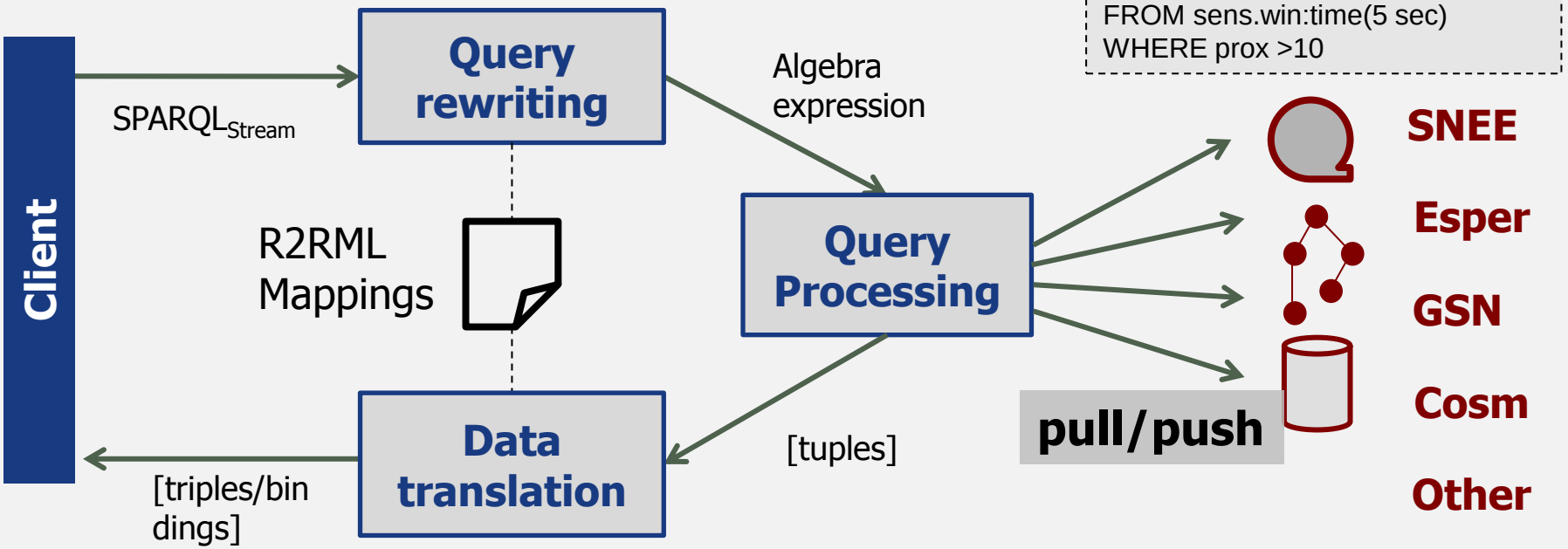
- *Select* → 'SELECT' [*Xstream*] [*Distinct* | *Reduced*] …
- *Xstream* → 'RSTREAM' | 'ISTREAM' | 'DSTREAM'

Morph-streams: Overview

```
SELECT ?proximity
FROM STREAM
<http://streamreasoning.org/SensorReadings.srdf> [NOW–5
S]
WHERE {
  ?obs a ssn:ObservationValue;
       qudt:numericalValue ?proximity;
  FILTER (?proximity>10) }
```

$\pi_{timed,prox}$
$\sigma_{prox>10}$
$\omega_{5\ Seconds}$
sens

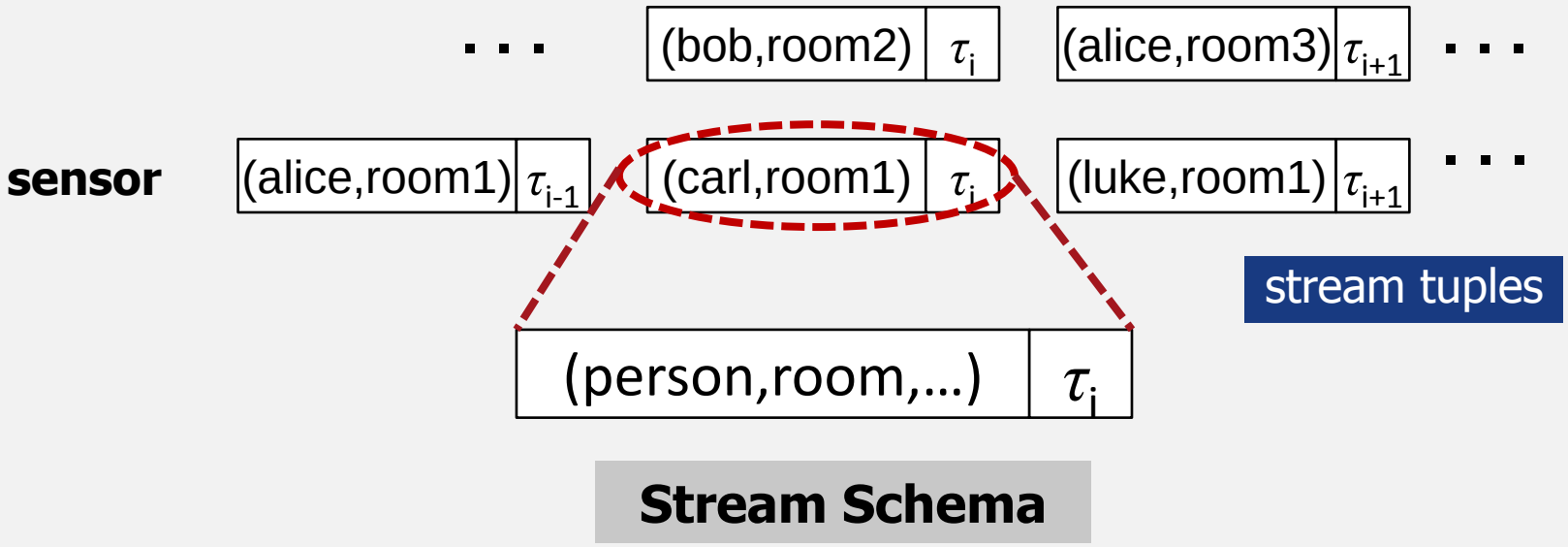
```
SELECT prox
FROM sens.win:time(5 sec)
WHERE prox >10
```



Morph-streams procesing SPARQL$_{Stream}$ queries

https://github.com/jpcik/morph-streams

Now, where is the data?



DSMS, CEP, middleware can evaluate queries over this model

Underlying Query Processors

Esper
- CEP/DSMS
- EPL language

```
SELECT prox FROM sensors.win:time(5 minute)
WHERE prox >10
```

SNEE
- DSMS/Sensor Network Query Evaluator
- Compile queries to sensor code

```
SELECT prox FROM sensors [FROM NOW-5
MINUTES TO NOW]
WHERE prox >10
```

GSN
- Sensor middleware
- REST API

```
http://montblanc.slf.ch:22001/multidata?vs[0]=sens
ors&
field[0]=proximity_field&c_min[0]=10&
from=15/05/2012+05:00:00&to=15/05/2012+10:00:
00
```

Cosm/Xively
- Sensor middleware
- Open platform
- REST API

```
http://api.cosm.com/v2/feeds/14321/datastreams/
4?start=2012-05-15T05:00:00Z&end=2012-05-
15T10:00:00Z
```

Underlying Query Processors

SPARQLStream

```
SELECT ?proximity
FROM STREAM <http://streamreasoning.org/SensorReadings.srdf>
[NOW–5 S]
WHERE {
  ?obs a ssn:ObservationValue;
       qudt:numericalValue ?proximity;
  FILTER (?proximity>10) }
```

R2RML

Query rewriting

$\pi_{timed,\ prox}$ | $\sigma_{prox>10}$ | $\omega_{5\ Seconds}$ | sensors

```
SELECT prox FROM sensors [FROM NOW-5 MINUTES TO NOW]
WHERE prox >10
```

SNEE (DSMS)

```
SELECT prox FROM sensors.win:time(5 minute)
WHERE prox >10
```

Esper (CEP)

```
http://montblanc.slf.ch:22001/multidata?vs[0]=sensors&field[0]=proximi
ty_field&c_min[0]=10&
from=15/05/2012+05:00:00&to=15/05/2012+10:00:00
```

GSN (middlwr)

```
http://api.cosm.com/v2/feeds/14321/datastreams/4?start=2012-05-
15T05:00:00Z&end=2012-05-15T10:00:00Z
```

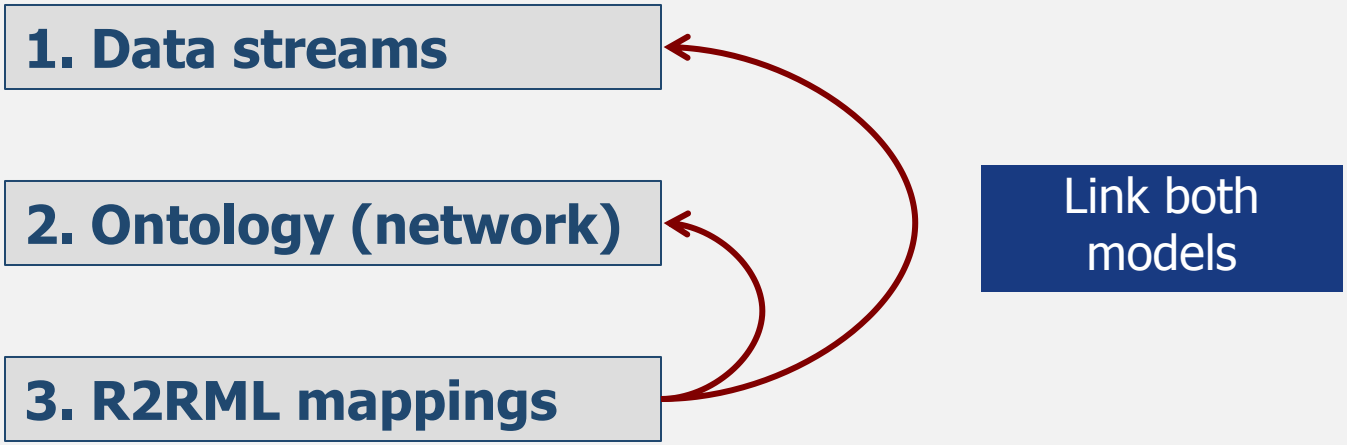
Cosm Xively

Underlying query processors

Features	Esper	SNEE	GSN	Cosm/Xively
Projection	✔	✔	✔	Fixed
Proj expression	✔	✔	✖	✖
Joins	✔	✔✖ only window	✖	✖
Union	✖	✔✖ not windows	✔	✖
Selection	✔	✔	✔	✖✔ limited
Aggregates	✔	✔	✔✖	✖
Time window	✔	✔	✔	✔
Tuple window	✔	✔	✔	✖
R2S	✔	✔	✖	✖
Conjunction, Disj	✔	✖	✖	✖
Repetition pattern	✔	✖	✖	✖
Sequence	✔	✖	✖	✖

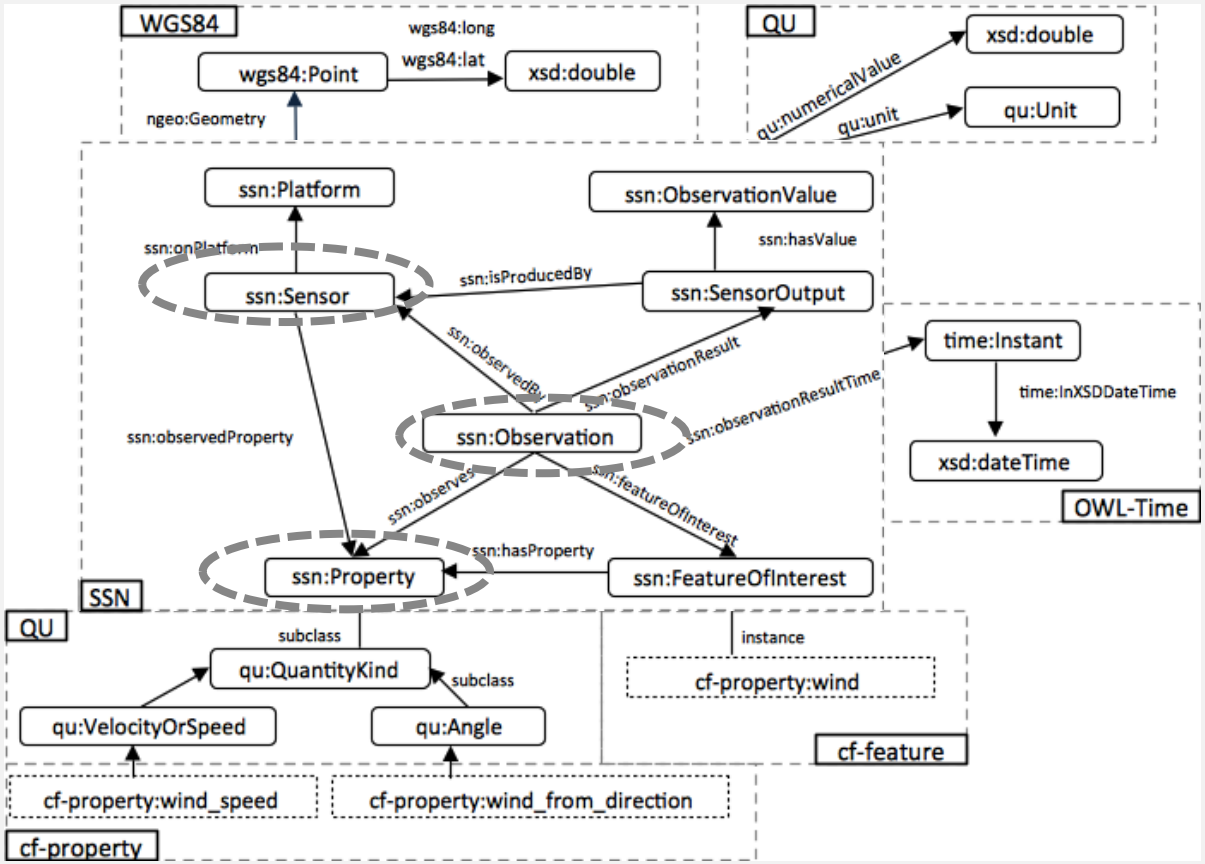
Configuring Morph-streams

- Main ingredients:



SSN Ontology with other ontologies

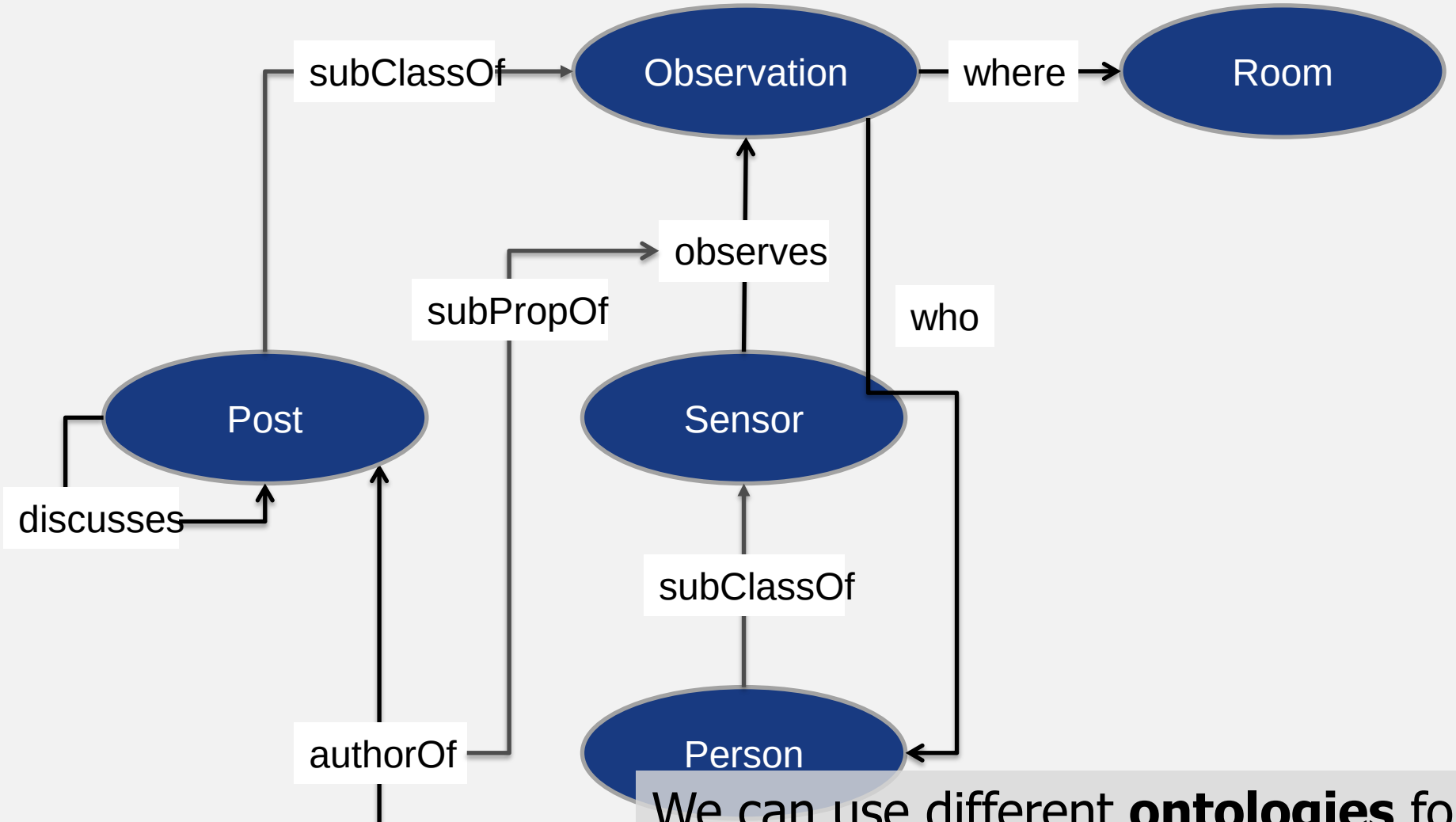
W3C SSN Ontology



modeling our streaming data

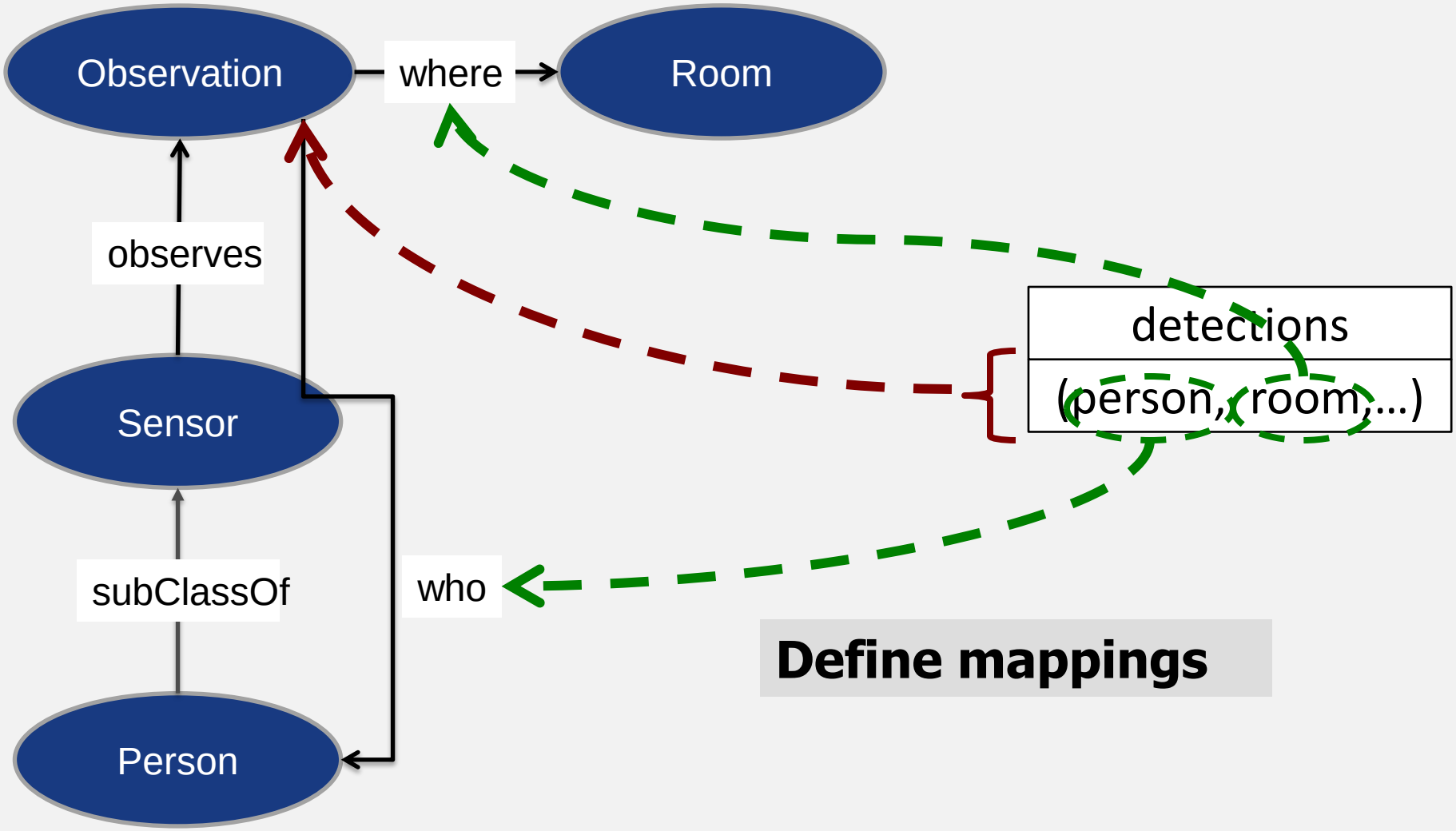
combine with **domain ontologies**

Our simpler ontology…

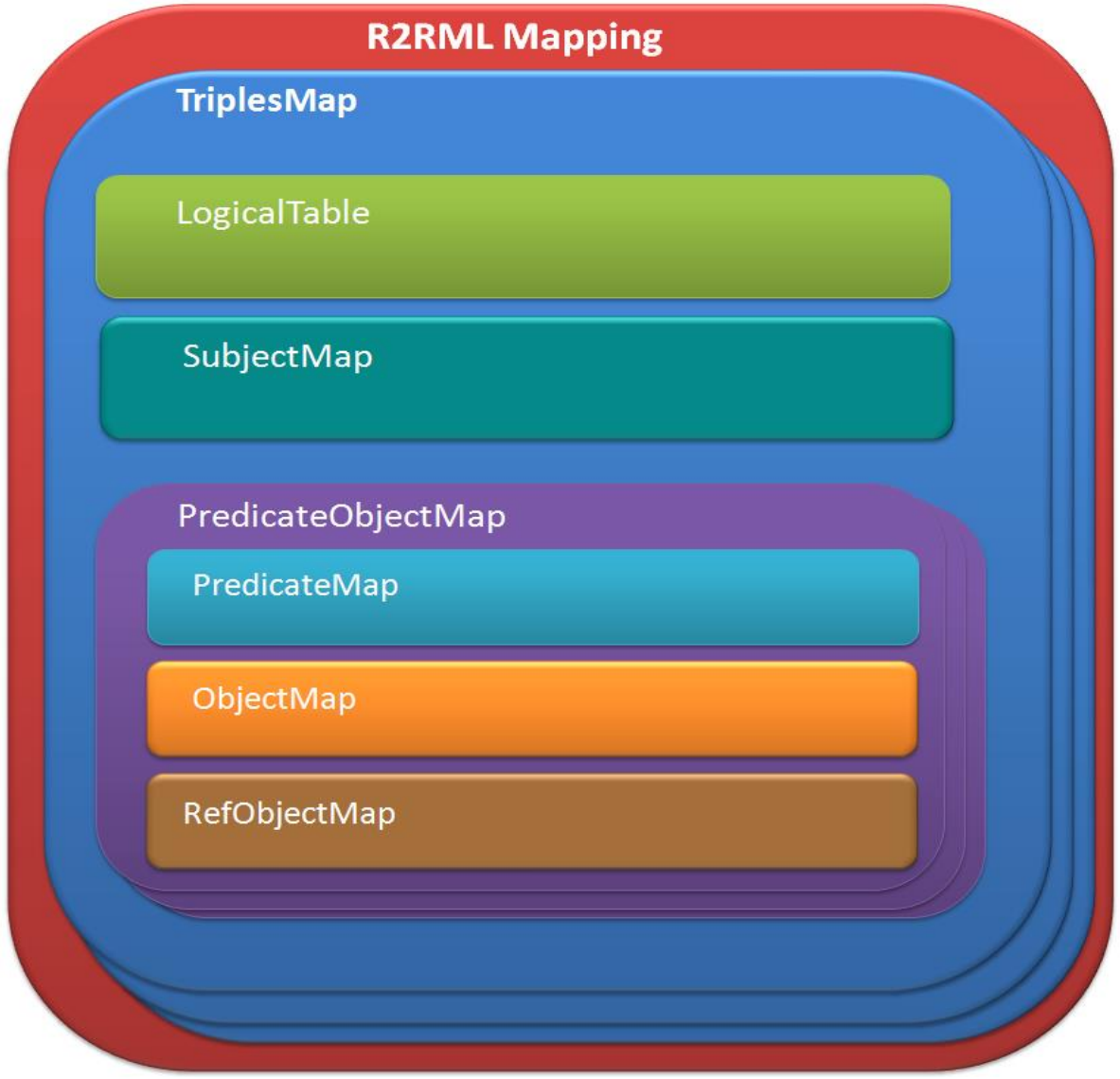


We can use different **ontologies** for the **same data**

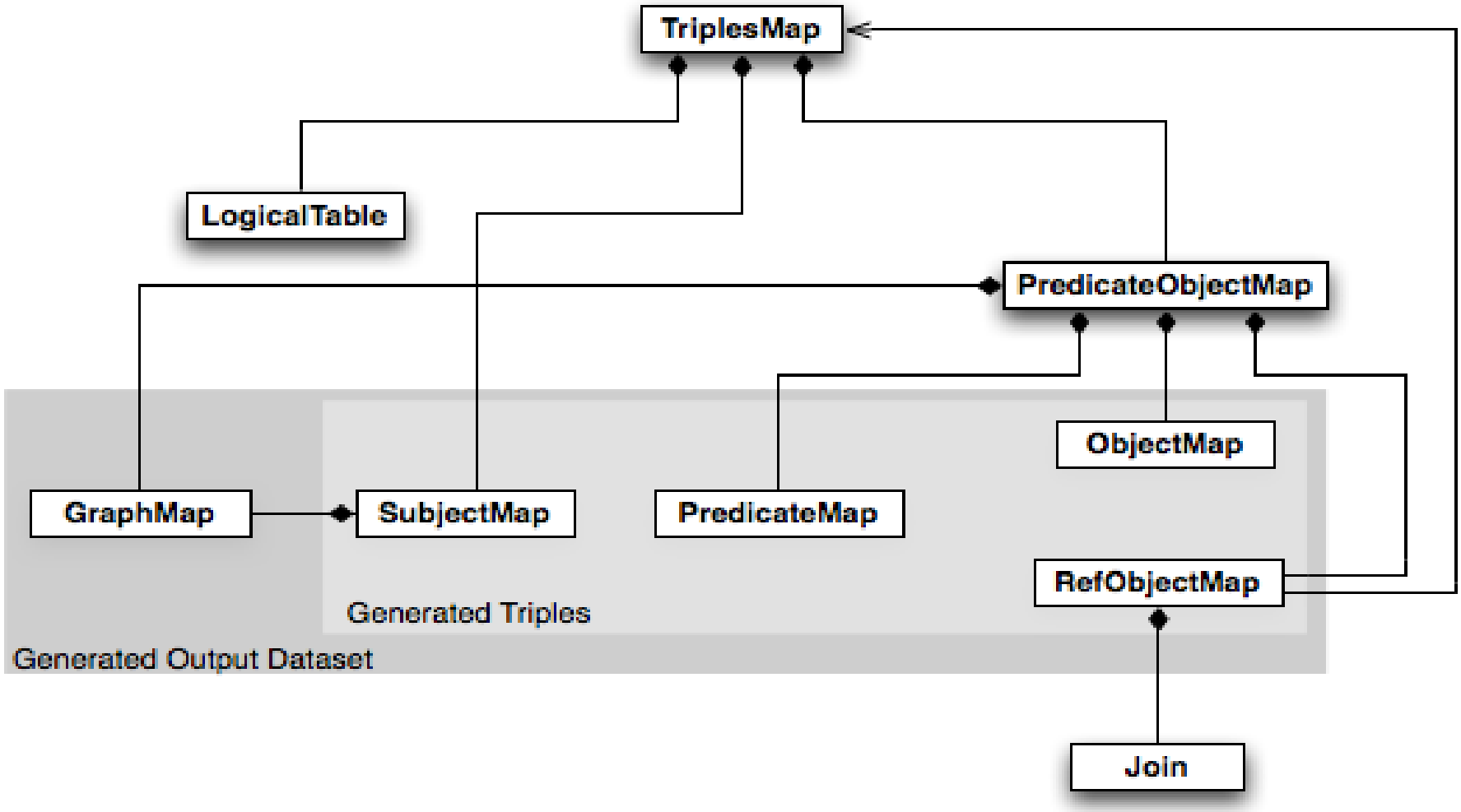
Our simpler ontology…



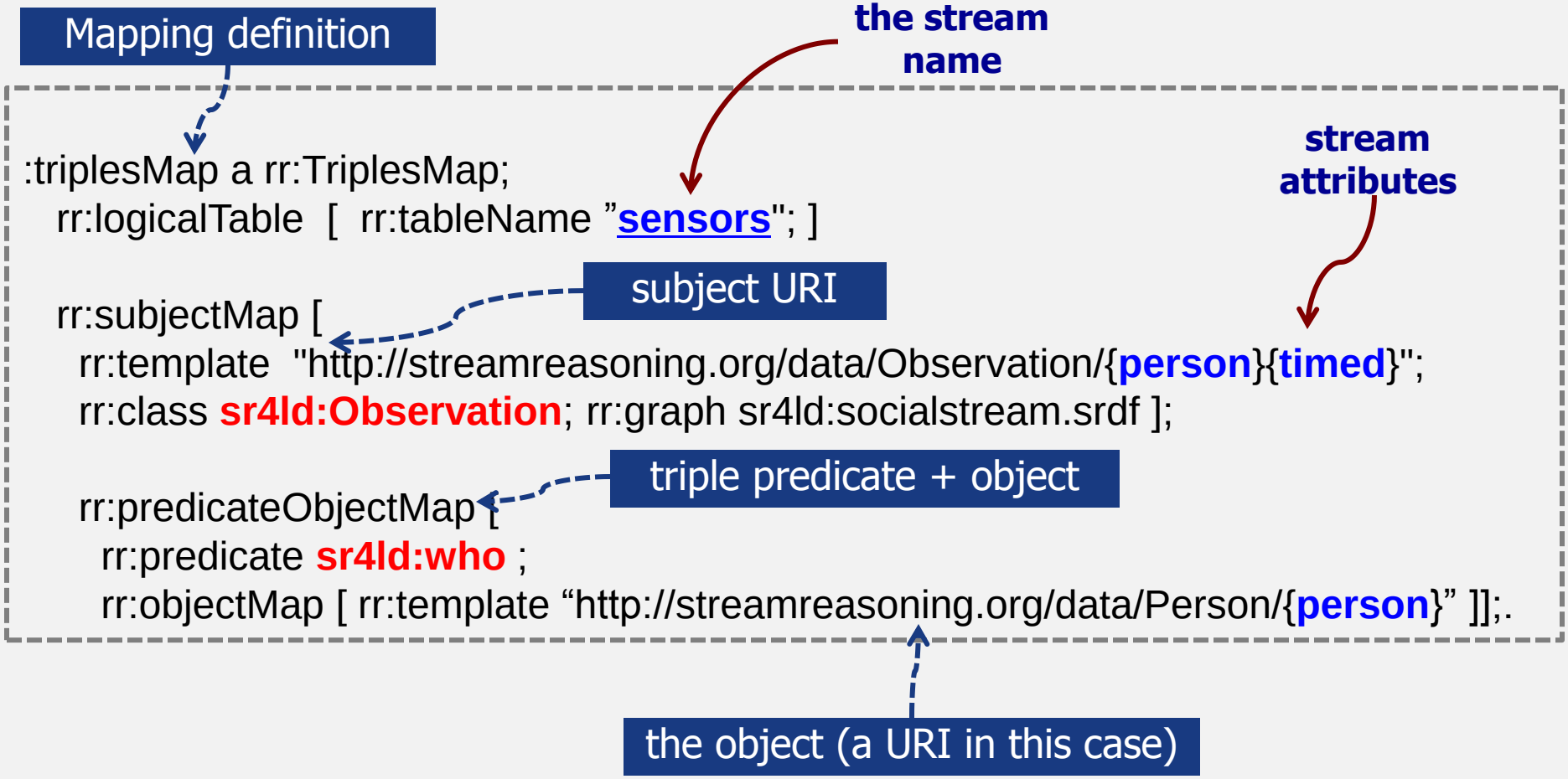
R2RML - Overview



R2RML - Overview



Encoding in R2RML



Now some code

Morph-streams:

- Coded in **Scala**
- JAR bundle, use it from Scala or **Java** code
- Maven, Sbt
- **Examples**
  - **One off query**
  - **Register continuous query**
  - **Pull data**
  - **Push**
  - **Basic REST**
- **https://github.com/jpcik/morph-streams**

Code examples

- Parse SPARQLStream

```
val query= “PREFIX sr4ld: <…>. SELECT ?a …”
val syntax= StreamQueryFactory.create(query);
```

- Execute One-off query

```
val query= “PREFIX sr4ld: <…>. SELECT ?a …”
mapping=Mapping(new URI(mappings/social.ttl))
val adapter:QueryEvaluator=Application.adapter(system)
val results= adapter.executeQuery(query,mapping)
```

Mapping

Bindings

Code examples

- Register and Pull

```
val queryid= adapter.registerQuery(query,mapping)
val results1=adapter.pull(queryid)
val results2=adapter.pull(queryid)
```

Query identifier

- Register and Push

Implement receiver

```
class ExampleReceiver extends StreamReceiver{
  override def receiveData(s:SparqlResults):Unit=
    Logger.debug("got: "+res)
 }
val receiver=new ExampleReceiver
val queryid= adapter.listenToQuery(query,mapping,receiver)
```

SPARQLStream from command line

- encoded_value=$(python -c "import urllib; print urllib.quote('"SELECT DISTINCT ?timeto ?obs FROM NAMED STREAM <http://emt.linkeddata.es/data#busstops.srdf> [NOW - 30 S] WHERE { ?obs a <http://emt.linkeddata.es/data#BusObservation>. ?obs <http://purl.oclc.org/NET/ssnx/ssn#observedBy><http://transporte.linkeddata.es/emt/busstop/id/2018>. ?obs <http://purl.oclc.org/NET/ssnx/ssn#observationResult> ?output. ?output <http://emt.linkeddata.es/data#timeToBusValue> ?av. ?av <http://data.nasa.gov/qudt/owl/qudt#numericValue> ?timeto. }'")")

Just encoding query

- curl "http://streams.linkeddata.es/emt/sparqlstream?query=$encoded_value"

Disclaimer: Simplistic, not implementing all of the SPARQL protocol

Sample result


```
{
  "head": {
    "vars": [ "timeto" , "obs" ]
  } ,
  "results": {
    "bindings": [
      {
        "timeto": { "datatype": "http://www.w3.org/2001/XMLSchema#string" , "type": "typed-literal" , "value": "0" } ,
        "obs": { "type": "uri" , "value": "http://transporte.linkeddata.es/emt/busstop/id/2018/busline/9/observation/20/09/2013%2010:28:19%20%2B0200" }
      }
    ]
  }
}
```


Bindings in JSON

Resources

- Morph-Streams
  - https://github.com/jpcik/morph-streams
- See demos
  - http://transporte.linkeddata.es/ (SPARQL-Stream tab)
  - Check our Madrid buses demo at SSN2013 workshop tomorrow
- Read out more
  - Enabling Query Technologies for the Semantic Sensor Web. J.-P. Calbimonte, H. Jeung, O. Corcho and K. Aberer. International Journal on Semantic Web and Information Systems IJSWIS, Volume 8(1)., 2012
- Contact point
  - jp.calbimonte@upm.es
  - ocorcho@fi.upm.es

Tutorial on RDF Stream Processing

M. Balduini, J-P Calbimonte, O. Corcho,
D. Dell'Aglio, E. Della Valle

http://streamreasoning.org/rsp2014

SPARQLStream: Ontology-based access to data streams

Jean Paul Calbimonte, Oscar Corcho
jp.calbimonte@upm.es, ocorcho@fi.upm.es
http://www.oeg-upm.net/

RDF Streams

$\langle s,p,o \rangle$

W3C RDF

<aemet:observation1, ssn:observedBy, aemet:Sensor3>

<aemet:observation1, qudt:hasNumericValue, “15.5”>

For streams?

$(\langle s,p,o \rangle,\tau)$

(<aemet:observation1, qudt:hasNumericValue, “15.5”>,**34532**)

timestamped triples

- Gutierrez et al. (2007) Introducing time into RDF. IEEE TKDE
- Rodríguez et al. (2009) Semantic management of streaming data. SSN